

libstdc++

Generated by Doxygen 1.8.11

Contents

1	Mathematical Special Functions	2
1.1	Introduction and History	2
1.2	Contents	2
1.3	General Features	3
1.3.1	Argument Promotion	3
1.3.2	NaN Arguments	3
1.4	Implementation	3
1.5	Testing	3
1.6	General Bibliography	4
2	Todo List	4
3	Module Documentation	5
3.1	Adaptors for pointers to functions	6
3.1.1	Detailed Description	6
3.1.2	Function Documentation	6
3.2	Adaptors for pointers to members	7
3.2.1	Detailed Description	7
3.3	Algorithms	8
3.3.1	Detailed Description	8
3.4	Allocators	9
3.4.1	Detailed Description	10
3.4.2	Typedef Documentation	10
3.5	Arithmetic Classes	11
3.5.1	Detailed Description	11
3.6	Array creation functions	12
3.6.1	Detailed Description	12
3.7	Associative	13

3.7.1 Detailed Description	13
3.8 Atomics	14
3.8.1 Detailed Description	18
3.8.2 Macro Definition Documentation	18
3.8.3 Typedef Documentation	18
3.8.4 Enumeration Type Documentation	23
3.8.5 Function Documentation	23
3.9 Base and Implementation Classes	24
3.9.1 Detailed Description	25
3.9.2 Enumeration Type Documentation	25
3.10 Base and Implementation Classes	26
3.10.1 Detailed Description	28
3.11 Base and Policy Classes	29
3.11.1 Detailed Description	29
3.12 Base and Policy Classes	30
3.12.1 Detailed Description	30
3.13 Base and Policy Classes	31
3.13.1 Detailed Description	31
3.14 Bernoulli Distributions	32
3.14.1 Detailed Description	33
3.14.2 Function Documentation	33
3.15 Binary Search	36
3.15.1 Detailed Description	36
3.15.2 Function Documentation	37
3.16 Binder Classes	41
3.16.1 Detailed Description	42
3.16.2 Function Documentation	42
3.17 Boolean Operations Classes	43

3.17.1 Detailed Description	43
3.18 Branch-Based	44
3.18.1 Detailed Description	44
3.19 Comparison Classes	45
3.19.1 Detailed Description	45
3.20 Complex Numbers	46
3.20.1 Detailed Description	49
3.20.2 Function Documentation	49
3.21 Concurrency	57
3.21.1 Detailed Description	57
3.22 Condition Variables	58
3.22.1 Detailed Description	58
3.22.2 Enumeration Type Documentation	58
3.23 Const-propagating wrapper	59
3.23.1 Detailed Description	60
3.24 Containers	61
3.24.1 Detailed Description	61
3.25 Containers	62
3.25.1 Detailed Description	62
3.26 Data Structure Type	63
3.26.1 Detailed Description	63
3.27 Decimal Floating-Point Arithmetic	64
3.27.1 Detailed Description	64
3.28 Diagnostics	65
3.28.1 Detailed Description	65
3.29 Dynamic Bitset.	66
3.29.1 Detailed Description	67
3.29.2 Function Documentation	67

3.30 Exceptions	71
3.30.1 Detailed Description	71
3.31 Exceptions	72
3.31.1 Detailed Description	73
3.31.2 Typedef Documentation	74
3.31.3 Function Documentation	74
3.32 Experimental	76
3.32.1 Detailed Description	76
3.33 Extensions	77
3.33.1 Detailed Description	77
3.34 Filesystem	78
3.34.1 Detailed Description	81
3.34.2 Enumeration Type Documentation	82
3.35 Function Objects	83
3.35.1 Detailed Description	84
3.35.2 Function Documentation	84
3.36 Futures	85
3.36.1 Detailed Description	86
3.36.2 Enumeration Type Documentation	86
3.36.3 Function Documentation	87
3.37 Hash-Based	88
3.37.1 Detailed Description	88
3.38 Hashes	89
3.38.1 Detailed Description	89
3.39 Heap	90
3.39.1 Detailed Description	90
3.39.2 Function Documentation	90
3.40 Heap-Based	96

3.40.1 Detailed Description	97
3.40.2 Function Documentation	97
3.41 I/O	98
3.41.1 Detailed Description	99
3.41.2 Typedef Documentation	99
3.42 Invalidation Guarantees	103
3.42.1 Detailed Description	103
3.43 Iterator Tags	104
3.43.1 Detailed Description	104
3.44 Iterators	105
3.44.1 Detailed Description	108
3.44.2 Function Documentation	108
3.45 List-Based	111
3.45.1 Detailed Description	111
3.46 Locales	112
3.46.1 Detailed Description	112
3.46.2 Function Documentation	113
3.47 Mathematical Special Functions	115
3.47.1 Detailed Description	117
3.47.2 Function Documentation	117
3.48 Mathematical Special Functions	139
3.48.1 Detailed Description	141
3.48.2 Function Documentation	141
3.49 Memory	145
3.49.1 Detailed Description	145
3.50 Metaprogramming	146
3.50.1 Detailed Description	147
3.50.2 Typedef Documentation	147

3.51 Mutating	148
3.51.1 Detailed Description	150
3.51.2 Function Documentation	150
3.52 Mutexes	168
3.52.1 Detailed Description	168
3.52.2 Macro Definition Documentation	169
3.52.3 Typedef Documentation	169
3.52.4 Function Documentation	169
3.52.5 Variable Documentation	169
3.53 Negators	170
3.53.1 Detailed Description	170
3.53.2 Function Documentation	171
3.54 Non-Mutating	172
3.54.1 Detailed Description	173
3.54.2 Function Documentation	173
3.55 Normal Distributions	189
3.55.1 Detailed Description	190
3.55.2 Function Documentation	190
3.56 Numeric Arrays	192
3.56.1 Detailed Description	200
3.56.2 Function Documentation	200
3.57 Numerics	226
3.57.1 Detailed Description	226
3.58 Optional values	227
3.58.1 Detailed Description	230
3.58.2 Function Documentation	230
3.58.3 Variable Documentation	230
3.59 Pointer Abstractions	231

3.59.1 Detailed Description	234
3.59.2 Function Documentation	235
3.60 Poisson Distributions	246
3.60.1 Detailed Description	247
3.60.2 Function Documentation	247
3.61 Policy-Based Data Structures	252
3.61.1 Detailed Description	252
3.62 Random Number Distributions	253
3.62.1 Detailed Description	253
3.63 Random Number Generation	254
3.63.1 Detailed Description	254
3.63.2 Function Documentation	254
3.64 Random Number Generators	255
3.64.1 Detailed Description	256
3.64.2 Typedef Documentation	256
3.64.3 Function Documentation	257
3.65 Random Number Utilities	261
3.65.1 Detailed Description	261
3.66 Rational Arithmetic	262
3.66.1 Detailed Description	263
3.66.2 Typedef Documentation	263
3.67 Regular Expressions	264
3.67.1 Detailed Description	269
3.67.2 Typedef Documentation	269
3.67.3 Function Documentation	270
3.68 SGI	300
3.68.1 Detailed Description	301
3.68.2 Function Documentation	302

3.69 Sequences	309
3.69.1 Detailed Description	309
3.70 Set Operation	310
3.70.1 Detailed Description	310
3.70.2 Function Documentation	311
3.71 Sorting	316
3.71.1 Detailed Description	318
3.71.2 Function Documentation	318
3.72 Strings	335
3.72.1 Detailed Description	335
3.72.2 Typedef Documentation	335
3.73 Tags	337
3.73.1 Detailed Description	337
3.73.2 Typedef Documentation	337
3.74 Threads	338
3.74.1 Detailed Description	338
3.75 Time	339
3.75.1 Detailed Description	339
3.76 Traits	340
3.76.1 Detailed Description	341
3.77 Type-safe container of any type	342
3.77.1 Detailed Description	343
3.77.2 Function Documentation	343
3.78 Uniform Distributions	346
3.78.1 Detailed Description	346
3.78.2 Function Documentation	346
3.79 Unordered Associative	349
3.79.1 Detailed Description	349
3.80 Utilities	350
3.80.1 Detailed Description	353
3.80.2 Function Documentation	353
3.80.3 Variable Documentation	359

4 Namespace Documentation	360
4.1 __gnu_cxx Namespace Reference	360
4.1.1 Detailed Description	375
4.1.2 Function Documentation	375
4.2 __gnu_cxx::__detail Namespace Reference	388
4.2.1 Detailed Description	389
4.2.2 Function Documentation	389
4.3 __gnu_cxx::typelist Namespace Reference	390
4.3.1 Detailed Description	390
4.3.2 Function Documentation	390
4.4 __gnu_debug Namespace Reference	390
4.4.1 Detailed Description	397
4.4.2 Enumeration Type Documentation	397
4.4.3 Function Documentation	397
4.5 __gnu_internal Namespace Reference	401
4.5.1 Detailed Description	401
4.6 __gnu_parallel Namespace Reference	401
4.6.1 Detailed Description	409
4.6.2 Typedef Documentation	409
4.6.3 Enumeration Type Documentation	409
4.6.4 Function Documentation	410
4.6.5 Variable Documentation	450
4.7 __gnu_pbds Namespace Reference	451
4.7.1 Detailed Description	453
4.8 __gnu_profile Namespace Reference	453
4.8.1 Detailed Description	456
4.8.2 Typedef Documentation	457
4.8.3 Function Documentation	457

4.9	__gnu_sequential Namespace Reference	457
4.9.1	Detailed Description	457
4.10	abi Namespace Reference	458
4.10.1	Detailed Description	458
4.11	std Namespace Reference	458
4.11.1	Detailed Description	559
4.11.2	Typedef Documentation	559
4.11.3	Enumeration Type Documentation	561
4.11.4	Function Documentation	562
4.11.5	Variable Documentation	645
4.12	std::__debug Namespace Reference	647
4.12.1	Detailed Description	652
4.12.2	Function Documentation	652
4.13	std::__detail Namespace Reference	653
4.13.1	Detailed Description	655
4.13.2	Function Documentation	655
4.14	std::__parallel Namespace Reference	656
4.14.1	Detailed Description	674
4.15	std::__profile Namespace Reference	674
4.15.1	Detailed Description	679
4.15.2	Function Documentation	679
4.16	std::chrono Namespace Reference	679
4.16.1	Detailed Description	681
4.16.2	Typedef Documentation	681
4.16.3	Function Documentation	682
4.17	std::decimal Namespace Reference	683
4.17.1	Detailed Description	691
4.17.2	Function Documentation	692

4.18	std::placeholders Namespace Reference	692
4.18.1	Detailed Description	692
4.19	std::regex_constants Namespace Reference	693
4.19.1	Detailed Description	694
4.19.2	Enumeration Type Documentation	694
4.19.3	Function Documentation	695
4.19.4	Variable Documentation	700
4.20	std::rel_ops Namespace Reference	705
4.20.1	Detailed Description	705
4.20.2	Function Documentation	705
4.21	std::this_thread Namespace Reference	707
4.21.1	Detailed Description	707
4.21.2	Function Documentation	707
4.22	std::tr1 Namespace Reference	708
4.22.1	Detailed Description	710
4.23	std::tr1::__detail Namespace Reference	710
4.23.1	Detailed Description	710
4.24	std::tr2 Namespace Reference	711
4.24.1	Detailed Description	712
4.25	std::tr2::__detail Namespace Reference	712
4.25.1	Detailed Description	712

5	Class Documentation	712
5.1	__cxxabiv1::__forced_unwind Class Reference	712
5.1.1	Detailed Description	712
5.2	__gnu_cxx::__alloc_traits<_Alloc> Struct Template Reference	713
5.2.1	Detailed Description	715
5.2.2	Member Typedef Documentation	715
5.2.3	Member Function Documentation	716
5.3	__gnu_cxx::__common_pool_policy<_PoolTp, _Thread> Struct Template Reference	719
5.3.1	Detailed Description	719
5.4	__gnu_cxx::__detail::__mini_vector<_Tp> Class Template Reference	719
5.4.1	Detailed Description	720
5.5	__gnu_cxx::__detail::__Bitmap_counter<_Tp> Class Template Reference	720
5.5.1	Detailed Description	720
5.6	__gnu_cxx::__detail::__Ffit_finder<_Tp> Class Template Reference	721
5.6.1	Detailed Description	721
5.6.2	Member Typedef Documentation	721
5.7	__gnu_cxx::__mt_alloc<_Tp, _Poolp> Class Template Reference	722
5.7.1	Detailed Description	723
5.8	__gnu_cxx::__mt_alloc_base<_Tp> Class Template Reference	723
5.8.1	Detailed Description	724
5.9	__gnu_cxx::__per_type_pool_policy<_Tp, _PoolTp, _Thread> Struct Template Reference	724
5.9.1	Detailed Description	724
5.10	__gnu_cxx::__pool<_Thread> Class Template Reference	725
5.10.1	Detailed Description	725
5.11	__gnu_cxx::__pool<false> Class Template Reference	725
5.11.1	Detailed Description	726
5.12	__gnu_cxx::__pool<true> Class Template Reference	726
5.12.1	Detailed Description	727

5.13	__gnu_cxx::__pool_alloc< _Tp > Class Template Reference	727
5.13.1	Detailed Description	729
5.14	__gnu_cxx::__pool_alloc_base Class Reference	729
5.14.1	Detailed Description	730
5.15	__gnu_cxx::__pool_base Struct Reference	730
5.15.1	Detailed Description	731
5.16	__gnu_cxx::__rc_string_base< _CharT, _Traits, _Alloc > Class Template Reference	731
5.16.1	Detailed Description	733
5.17	__gnu_cxx::__scoped_lock Class Reference	734
5.17.1	Detailed Description	734
5.18	__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base > Class Template Reference	734
5.18.1	Detailed Description	738
5.18.2	Constructor & Destructor Documentation	738
5.18.3	Member Function Documentation	741
5.18.4	Member Data Documentation	793
5.19	__gnu_cxx::__Caster< _ToType > Struct Template Reference	794
5.19.1	Detailed Description	794
5.20	__gnu_cxx::__Char_types< _CharT > Struct Template Reference	794
5.20.1	Detailed Description	794
5.21	__gnu_cxx::__ExtPtr_allocator< _Tp > Class Template Reference	795
5.21.1	Detailed Description	796
5.22	__gnu_cxx::__Invalid_type Struct Reference	796
5.22.1	Detailed Description	796
5.23	__gnu_cxx::__Pointer_adapter< _Storage_policy > Class Template Reference	796
5.23.1	Detailed Description	798
5.24	__gnu_cxx::__Relative_pointer_impl< _Tp > Class Template Reference	798
5.24.1	Detailed Description	799
5.25	__gnu_cxx::__Relative_pointer_impl< const _Tp > Class Template Reference	799

5.25.1 Detailed Description	799
5.26 <code>__gnu_cxx::Std_pointer_impl<_Tp></code> Class Template Reference	800
5.26.1 Detailed Description	800
5.27 <code>__gnu_cxx::Unqualified_type<_Tp></code> Struct Template Reference	800
5.27.1 Detailed Description	800
5.28 <code>__gnu_cxx::annotate_base</code> Struct Reference	801
5.28.1 Detailed Description	801
5.29 <code>__gnu_cxx::array_allocator<_Tp, _Array></code> Class Template Reference	802
5.29.1 Detailed Description	803
5.30 <code>__gnu_cxx::array_allocator_base<_Tp></code> Class Template Reference	803
5.30.1 Detailed Description	804
5.31 <code>__gnu_cxx::binary_compose<_Operation1, _Operation2, _Operation3></code> Class Template Reference	804
5.31.1 Detailed Description	805
5.31.2 Member Typedef Documentation	805
5.32 <code>__gnu_cxx::bitmap_allocator<_Tp></code> Class Template Reference	806
5.32.1 Detailed Description	807
5.32.2 Member Function Documentation	807
5.33 <code>__gnu_cxx::char_traits<_CharT></code> Struct Template Reference	808
5.33.1 Detailed Description	809
5.34 <code>__gnu_cxx::character<_Value, _Int, _St></code> Struct Template Reference	809
5.34.1 Detailed Description	810
5.35 <code>__gnu_cxx::condition_base</code> Struct Reference	810
5.35.1 Detailed Description	810
5.36 <code>__gnu_cxx::constant_binary_fun<_Result, _Arg1, _Arg2></code> Struct Template Reference	811
5.36.1 Detailed Description	811
5.37 <code>__gnu_cxx::constant_unary_fun<_Result, _Argument></code> Struct Template Reference	811
5.37.1 Detailed Description	812
5.38 <code>__gnu_cxx::constant_void_fun<_Result></code> Struct Template Reference	812

5.38.1 Detailed Description	813
5.39 __gnu_cxx::debug_allocator<_Alloc> Class Template Reference	813
5.39.1 Detailed Description	814
5.40 __gnu_cxx::enc_filebuf<_CharT> Class Template Reference	814
5.40.1 Detailed Description	817
5.40.2 Member Function Documentation	817
5.40.3 Member Data Documentation	832
5.41 __gnu_cxx::encoding_char_traits<_CharT> Struct Template Reference	836
5.41.1 Detailed Description	837
5.42 __gnu_cxx::encoding_state Class Reference	837
5.42.1 Detailed Description	838
5.43 __gnu_cxx::forced_error Struct Reference	838
5.43.1 Detailed Description	838
5.43.2 Member Function Documentation	838
5.44 __gnu_cxx::free_list Class Reference	839
5.44.1 Detailed Description	839
5.44.2 Member Function Documentation	839
5.45 __gnu_cxx::hash_map<_Key, _Tp, _HashFn, _EqualKey, _Alloc> Class Template Reference	840
5.45.1 Detailed Description	842
5.46 __gnu_cxx::hash_multimap<_Key, _Tp, _HashFn, _EqualKey, _Alloc> Class Template Reference	842
5.46.1 Detailed Description	844
5.47 __gnu_cxx::hash_multiset<_Value, _HashFcn, _EqualKey, _Alloc> Class Template Reference	844
5.47.1 Detailed Description	845
5.48 __gnu_cxx::hash_set<_Value, _HashFcn, _EqualKey, _Alloc> Class Template Reference	846
5.48.1 Detailed Description	847
5.49 __gnu_cxx::limit_condition Struct Reference	847
5.49.1 Detailed Description	848
5.50 __gnu_cxx::limit_condition::always_adjustor Struct Reference	848

5.50.1 Detailed Description	848
5.51 __gnu_cxx::limit_condition::limit_adjustor Struct Reference	848
5.51.1 Detailed Description	849
5.52 __gnu_cxx::limit_condition::never_adjustor Struct Reference	849
5.52.1 Detailed Description	849
5.53 __gnu_cxx::malloc_allocator< _Tp > Class Template Reference	849
5.53.1 Detailed Description	850
5.54 __gnu_cxx::new_allocator< _Tp > Class Template Reference	850
5.54.1 Detailed Description	851
5.55 __gnu_cxx::project1st< _Arg1, _Arg2 > Struct Template Reference	852
5.55.1 Detailed Description	852
5.55.2 Member Typedef Documentation	852
5.56 __gnu_cxx::project2nd< _Arg1, _Arg2 > Struct Template Reference	853
5.56.1 Detailed Description	853
5.56.2 Member Typedef Documentation	853
5.57 __gnu_cxx::random_condition Struct Reference	854
5.57.1 Detailed Description	854
5.58 __gnu_cxx::random_condition::always_adjustor Struct Reference	854
5.58.1 Detailed Description	855
5.59 __gnu_cxx::random_condition::group_adjustor Struct Reference	855
5.59.1 Detailed Description	855
5.60 __gnu_cxx::random_condition::never_adjustor Struct Reference	855
5.60.1 Detailed Description	855
5.61 __gnu_cxx::rb_tree< _Key, _Value, _KeyOfValue, _Compare, _Alloc > Struct Template Reference	856
5.61.1 Detailed Description	859
5.62 __gnu_cxx::recursive_init_error Class Reference	859
5.62.1 Detailed Description	860
5.62.2 Member Function Documentation	860

5.63	__gnu_cxx::rope<_CharT, _Alloc> Class Template Reference	860
5.63.1	Detailed Description	865
5.64	__gnu_cxx::select1st<_Pair> Struct Template Reference	865
5.64.1	Detailed Description	866
5.64.2	Member Typedef Documentation	866
5.65	__gnu_cxx::select2nd<_Pair> Struct Template Reference	866
5.65.1	Detailed Description	867
5.65.2	Member Typedef Documentation	867
5.66	__gnu_cxx::slist<_Tp, _Alloc> Class Template Reference	867
5.66.1	Detailed Description	869
5.67	__gnu_cxx::stdio_filebuf<_CharT, _Traits> Class Template Reference	870
5.67.1	Detailed Description	873
5.67.2	Constructor & Destructor Documentation	873
5.67.3	Member Function Documentation	874
5.67.4	Member Data Documentation	892
5.68	__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits> Class Template Reference	896
5.68.1	Detailed Description	898
5.68.2	Member Function Documentation	898
5.68.3	Member Data Documentation	913
5.69	__gnu_cxx::subtractive_rng Class Reference	914
5.69.1	Detailed Description	915
5.69.2	Member Typedef Documentation	915
5.69.3	Constructor & Destructor Documentation	915
5.69.4	Member Function Documentation	915
5.70	__gnu_cxx::temporary_buffer<_ForwardIterator, _Tp> Struct Template Reference	916
5.70.1	Detailed Description	917
5.70.2	Constructor & Destructor Documentation	917
5.70.3	Member Function Documentation	917

5.71	__gnu_cxx::throw_allocator_base< _Tp, _Cond > Class Template Reference	918
5.71.1	Detailed Description	919
5.72	__gnu_cxx::throw_allocator_limit< _Tp > Struct Template Reference	920
5.72.1	Detailed Description	921
5.73	__gnu_cxx::throw_allocator_random< _Tp > Struct Template Reference	922
5.73.1	Detailed Description	923
5.74	__gnu_cxx::throw_value_base< _Cond > Struct Template Reference	923
5.74.1	Detailed Description	924
5.75	__gnu_cxx::throw_value_limit Struct Reference	925
5.75.1	Detailed Description	926
5.76	__gnu_cxx::throw_value_random Struct Reference	926
5.76.1	Detailed Description	927
5.77	__gnu_cxx::unary_compose< _Operation1, _Operation2 > Class Template Reference	927
5.77.1	Detailed Description	928
5.77.2	Member Typedef Documentation	928
5.78	__gnu_debug::After_nth_from< _Iterator > Class Template Reference	929
5.78.1	Detailed Description	929
5.79	__gnu_debug::BeforeBeginHelper< _Sequence > Struct Template Reference	929
5.79.1	Detailed Description	929
5.80	__gnu_debug::Equal_to< _Type > Class Template Reference	930
5.80.1	Detailed Description	930
5.81	__gnu_debug::Not_equal_to< _Type > Class Template Reference	930
5.81.1	Detailed Description	930
5.82	__gnu_debug::Safe_container< _SafeContainer, _Alloc, _SafeBase, _IsCxx11AllocatorAware > Class Template Reference	930
5.82.1	Detailed Description	931
5.83	__gnu_debug::Safe_forward_list< _SafeSequence > Class Template Reference	931
5.83.1	Detailed Description	932

5.83.2	Member Function Documentation	932
5.83.3	Member Data Documentation	934
5.84	<code>__gnu_debug::_Safe_iterator<_Iterator, _Sequence></code> Class Template Reference	934
5.84.1	Detailed Description	936
5.84.2	Constructor & Destructor Documentation	936
5.84.3	Member Function Documentation	938
5.84.4	Member Data Documentation	943
5.85	<code>__gnu_debug::_Safe_iterator_base</code> Class Reference	944
5.85.1	Detailed Description	945
5.85.2	Constructor & Destructor Documentation	945
5.85.3	Member Function Documentation	946
5.85.4	Member Data Documentation	948
5.86	<code>__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence></code> Class Template Reference	949
5.86.1	Detailed Description	950
5.86.2	Constructor & Destructor Documentation	951
5.86.3	Member Function Documentation	952
5.86.4	Member Data Documentation	958
5.87	<code>__gnu_debug::_Safe_local_iterator_base</code> Class Reference	959
5.87.1	Detailed Description	960
5.87.2	Constructor & Destructor Documentation	960
5.87.3	Member Function Documentation	961
5.87.4	Member Data Documentation	962
5.88	<code>__gnu_debug::_Safe_node_sequence<_Sequence></code> Class Template Reference	963
5.88.1	Detailed Description	964
5.88.2	Member Function Documentation	964
5.88.3	Member Data Documentation	966
5.89	<code>__gnu_debug::_Safe_sequence<_Sequence></code> Class Template Reference	967
5.89.1	Detailed Description	968

5.89.2	Member Function Documentation	968
5.89.3	Member Data Documentation	970
5.90	<code>__gnu_debug::__Safe_sequence_base</code> Class Reference	971
5.90.1	Detailed Description	972
5.90.2	Constructor & Destructor Documentation	972
5.90.3	Member Function Documentation	972
5.90.4	Member Data Documentation	974
5.91	<code>__gnu_debug::__Safe_unordered_container<_Container></code> Class Template Reference	974
5.91.1	Detailed Description	975
5.91.2	Member Function Documentation	976
5.91.3	Member Data Documentation	978
5.92	<code>__gnu_debug::__Safe_unordered_container_base</code> Class Reference	979
5.92.1	Detailed Description	980
5.92.2	Constructor & Destructor Documentation	980
5.92.3	Member Function Documentation	980
5.92.4	Member Data Documentation	982
5.93	<code>__gnu_debug::__Safe_vector<_SafeSequence, _BaseSequence></code> Class Template Reference	983
5.93.1	Detailed Description	983
5.94	<code>__gnu_debug::__Sequence_traits<_Sequence></code> Struct Template Reference	983
5.94.1	Detailed Description	984
5.95	<code>__gnu_debug::basic_string<_CharT, _Traits, _Allocator></code> Class Template Reference	984
5.95.1	Detailed Description	989
5.95.2	Member Function Documentation	989
5.95.3	Member Data Documentation	1009
5.96	<code>__gnu_parallel::__accumulate_binop_reduct<_BinOp></code> Struct Template Reference	1009
5.96.1	Detailed Description	1010
5.97	<code>__gnu_parallel::__accumulate_selector<_It></code> Struct Template Reference	1010
5.97.1	Detailed Description	1010

5.97.2	Member Function Documentation	1011
5.97.3	Member Data Documentation	1011
5.98	<code>__gnu_parallel::__adjacent_difference_selector< _It ></code> Struct Template Reference	1011
5.98.1	Detailed Description	1012
5.98.2	Member Data Documentation	1012
5.99	<code>__gnu_parallel::__adjacent_find_selector</code> Struct Reference	1012
5.99.1	Detailed Description	1013
5.99.2	Member Function Documentation	1013
5.100	<code>__gnu_parallel::__binder1st< _Operation, _FirstArgumentType, _SecondArgumentType, _ResultType ></code> Class Template Reference	1014
5.100.1	Detailed Description	1015
5.100.2	Member Typedef Documentation	1015
5.101	<code>__gnu_parallel::__binder2nd< _Operation, _FirstArgumentType, _SecondArgumentType, _ResultType ></code> Class Template Reference	1015
5.101.1	Detailed Description	1016
5.101.2	Member Typedef Documentation	1016
5.102	<code>__gnu_parallel::__count_if_selector< _It, _Diff ></code> Struct Template Reference	1017
5.102.1	Detailed Description	1017
5.102.2	Member Function Documentation	1017
5.102.3	Member Data Documentation	1018
5.103	<code>__gnu_parallel::__count_selector< _It, _Diff ></code> Struct Template Reference	1018
5.103.1	Detailed Description	1019
5.103.2	Member Function Documentation	1019
5.103.3	Member Data Documentation	1019
5.104	<code>__gnu_parallel::__fill_selector< _It ></code> Struct Template Reference	1020
5.104.1	Detailed Description	1020
5.104.2	Member Function Documentation	1020
5.104.3	Member Data Documentation	1021
5.105	<code>__gnu_parallel::__find_first_of_selector< _Filterator ></code> Struct Template Reference	1021

5.105.1 Detailed Description	1022
5.105.2 Member Function Documentation	1022
5.106 <code>__gnu_parallel::__find_if_selector</code> Struct Reference	1023
5.106.1 Detailed Description	1023
5.106.2 Member Function Documentation	1023
5.107 <code>__gnu_parallel::__for_each_selector<_It></code> Struct Template Reference	1024
5.107.1 Detailed Description	1025
5.107.2 Member Function Documentation	1025
5.107.3 Member Data Documentation	1025
5.108 <code>__gnu_parallel::__generate_selector<_It></code> Struct Template Reference	1026
5.108.1 Detailed Description	1026
5.108.2 Member Function Documentation	1026
5.108.3 Member Data Documentation	1027
5.109 <code>__gnu_parallel::__generic_find_selector</code> Struct Reference	1027
5.109.1 Detailed Description	1028
5.110 <code>__gnu_parallel::__generic_for_each_selector<_It></code> Struct Template Reference	1029
5.110.1 Detailed Description	1030
5.110.2 Member Data Documentation	1030
5.111 <code>__gnu_parallel::__identity_selector<_It></code> Struct Template Reference	1030
5.111.1 Detailed Description	1031
5.111.2 Member Function Documentation	1031
5.111.3 Member Data Documentation	1031
5.112 <code>__gnu_parallel::__inner_product_selector<_It, _It2, _Tp></code> Struct Template Reference	1032
5.112.1 Detailed Description	1032
5.112.2 Constructor & Destructor Documentation	1032
5.112.3 Member Function Documentation	1033
5.112.4 Member Data Documentation	1033
5.113 <code>__gnu_parallel::__max_element_reduct<_Compare, _It></code> Struct Template Reference	1034

5.113.1 Detailed Description	1034
5.114 <code>__gnu_parallel::__min_element_reduct< _Compare, _It ></code> Struct Template Reference	1034
5.114.1 Detailed Description	1035
5.115 <code>__gnu_parallel::__mismatch_selector</code> Struct Reference	1035
5.115.1 Detailed Description	1036
5.115.2 Member Function Documentation	1036
5.116 <code>__gnu_parallel::__multiway_merge_3_variant_sentinel_switch< __sentinels, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare ></code> Struct Template Reference	1036
5.116.1 Detailed Description	1037
5.117 <code>__gnu_parallel::__multiway_merge_3_variant_sentinel_switch< true, _RAIterIterator, _RAIter3, _↵ DifferenceTp, _Compare ></code> Struct Template Reference	1037
5.117.1 Detailed Description	1037
5.118 <code>__gnu_parallel::__multiway_merge_4_variant_sentinel_switch< __sentinels, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare ></code> Struct Template Reference	1037
5.118.1 Detailed Description	1038
5.119 <code>__gnu_parallel::__multiway_merge_4_variant_sentinel_switch< true, _RAIterIterator, _RAIter3, _↵ DifferenceTp, _Compare ></code> Struct Template Reference	1038
5.119.1 Detailed Description	1038
5.120 <code>__gnu_parallel::__multiway_merge_k_variant_sentinel_switch< __sentinels, __stable, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare ></code> Struct Template Reference	1038
5.120.1 Detailed Description	1039
5.121 <code>__gnu_parallel::__multiway_merge_k_variant_sentinel_switch< false, __stable, _RAIterIterator, _RA↵ Iter3, _DifferenceTp, _Compare ></code> Struct Template Reference	1039
5.121.1 Detailed Description	1039
5.122 <code>__gnu_parallel::__replace_if_selector< _It, _Op, _Tp ></code> Struct Template Reference	1040
5.122.1 Detailed Description	1040
5.122.2 Constructor & Destructor Documentation	1040
5.122.3 Member Function Documentation	1041
5.122.4 Member Data Documentation	1041
5.123 <code>__gnu_parallel::__replace_selector< _It, _Tp ></code> Struct Template Reference	1042
5.123.1 Detailed Description	1042

5.123.2 Constructor & Destructor Documentation	1042
5.123.3 Member Function Documentation	1043
5.123.4 Member Data Documentation	1043
5.124 <code>__gnu_parallel::__transform1_selector<_It></code> Struct Template Reference	1044
5.124.1 Detailed Description	1044
5.124.2 Member Function Documentation	1044
5.124.3 Member Data Documentation	1045
5.125 <code>__gnu_parallel::__transform2_selector<_It></code> Struct Template Reference	1045
5.125.1 Detailed Description	1046
5.125.2 Member Function Documentation	1046
5.125.3 Member Data Documentation	1046
5.126 <code>__gnu_parallel::__unary_negate<_Predicate, argument_type></code> Class Template Reference	1047
5.126.1 Detailed Description	1047
5.126.2 Member Typedef Documentation	1048
5.127 <code>__gnu_parallel::_DRandomShufflingGlobalData<_RAIter></code> Struct Template Reference	1048
5.127.1 Detailed Description	1048
5.127.2 Constructor & Destructor Documentation	1049
5.127.3 Member Data Documentation	1049
5.128 <code>__gnu_parallel::_DRSSorterPU<_RAIter, _RandomNumberGenerator></code> Struct Template Reference	1050
5.128.1 Detailed Description	1050
5.128.2 Member Data Documentation	1051
5.129 <code>__gnu_parallel::_DummyReduct</code> Struct Reference	1052
5.129.1 Detailed Description	1052
5.130 <code>__gnu_parallel::_EqualFromLess<_T1, _T2, _Compare></code> Class Template Reference	1052
5.130.1 Detailed Description	1053
5.130.2 Member Typedef Documentation	1053
5.131 <code>__gnu_parallel::_EqualTo<_T1, _T2></code> Struct Template Reference	1053
5.131.1 Detailed Description	1054

5.131.2 Member Typedef Documentation	1054
5.132__gnu_parallel::_GuardedIterator< _RAIter, _Compare > Class Template Reference	1055
5.132.1 Detailed Description	1055
5.132.2 Constructor & Destructor Documentation	1055
5.132.3 Member Function Documentation	1056
5.132.4 Friends And Related Function Documentation	1056
5.133__gnu_parallel::_IteratorPair< _Iterator1, _Iterator2, _IteratorCategory > Class Template Reference	1057
5.133.1 Detailed Description	1058
5.133.2 Member Typedef Documentation	1058
5.133.3 Member Data Documentation	1059
5.134__gnu_parallel::_IteratorTriple< _Iterator1, _Iterator2, _Iterator3, _IteratorCategory > Class Template Reference	1059
5.134.1 Detailed Description	1060
5.135__gnu_parallel::_Job< _DifferenceTp > Struct Template Reference	1060
5.135.1 Detailed Description	1061
5.135.2 Member Data Documentation	1061
5.136__gnu_parallel::_Less< _T1, _T2 > Struct Template Reference	1062
5.136.1 Detailed Description	1062
5.136.2 Member Typedef Documentation	1062
5.137__gnu_parallel::_Lexicographic< _T1, _T2, _Compare > Class Template Reference	1063
5.137.1 Detailed Description	1064
5.137.2 Member Typedef Documentation	1064
5.138__gnu_parallel::_LexicographicReverse< _T1, _T2, _Compare > Class Template Reference	1064
5.138.1 Detailed Description	1065
5.138.2 Member Typedef Documentation	1065
5.139__gnu_parallel::_LoserTree< __stable, _Tp, _Compare > Class Template Reference	1066
5.139.1 Detailed Description	1066
5.139.2 Member Function Documentation	1067

5.139.3 Member Data Documentation	1067
5.140__gnu_parallel::_LoserTree< false, _Tp, _Compare > Class Template Reference	1068
5.140.1 Detailed Description	1069
5.140.2 Member Function Documentation	1069
5.140.3 Member Data Documentation	1071
5.141__gnu_parallel::_LoserTreeBase< _Tp, _Compare > Class Template Reference	1072
5.141.1 Detailed Description	1072
5.141.2 Constructor & Destructor Documentation	1073
5.141.3 Member Function Documentation	1073
5.141.4 Member Data Documentation	1074
5.142__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser Struct Reference	1075
5.142.1 Detailed Description	1075
5.142.2 Member Data Documentation	1075
5.143__gnu_parallel::_LoserTreePointer< __stable, _Tp, _Compare > Class Template Reference	1076
5.143.1 Detailed Description	1077
5.144__gnu_parallel::_LoserTreePointer< false, _Tp, _Compare > Class Template Reference	1077
5.144.1 Detailed Description	1078
5.145__gnu_parallel::_LoserTreePointerBase< _Tp, _Compare > Class Template Reference	1078
5.145.1 Detailed Description	1079
5.146__gnu_parallel::_LoserTreePointerBase< _Tp, _Compare >::_Loser Struct Reference	1079
5.146.1 Detailed Description	1080
5.147__gnu_parallel::_LoserTreePointerUnguarded< __stable, _Tp, _Compare > Class Template Reference	1080
5.147.1 Detailed Description	1081
5.148__gnu_parallel::_LoserTreePointerUnguarded< false, _Tp, _Compare > Class Template Reference	1081
5.148.1 Detailed Description	1082
5.149__gnu_parallel::_LoserTreePointerUnguardedBase< _Tp, _Compare > Class Template Reference	1082
5.149.1 Detailed Description	1083
5.150__gnu_parallel::_LoserTreeTraits< _Tp > Struct Template Reference	1083

5.150.1 Detailed Description	1083
5.150.2 Member Data Documentation	1084
5.151 <code>__gnu_parallel::_LoserTreeUnguarded< __stable, _Tp, _Compare ></code> Class Template Reference . . .	1084
5.151.1 Detailed Description	1085
5.152 <code>__gnu_parallel::_LoserTreeUnguarded< false, _Tp, _Compare ></code> Class Template Reference	1085
5.152.1 Detailed Description	1086
5.153 <code>__gnu_parallel::_LoserTreeUnguardedBase< _Tp, _Compare ></code> Class Template Reference	1086
5.153.1 Detailed Description	1087
5.154 <code>__gnu_parallel::_Multiplies< _Tp1, _Tp2, _Result ></code> Struct Template Reference	1087
5.154.1 Detailed Description	1088
5.154.2 Member Typedef Documentation	1088
5.155 <code>__gnu_parallel::_Nothing</code> Struct Reference	1089
5.155.1 Detailed Description	1089
5.155.2 Member Function Documentation	1089
5.156 <code>__gnu_parallel::_Piece< _DifferenceTp ></code> Struct Template Reference	1089
5.156.1 Detailed Description	1090
5.156.2 Member Data Documentation	1090
5.157 <code>__gnu_parallel::_Plus< _Tp1, _Tp2, _Result ></code> Struct Template Reference	1090
5.157.1 Detailed Description	1091
5.157.2 Member Typedef Documentation	1091
5.158 <code>__gnu_parallel::_PMWMSSortingData< _RAIter ></code> Struct Template Reference	1092
5.158.1 Detailed Description	1092
5.158.2 Member Data Documentation	1092
5.159 <code>__gnu_parallel::_PseudoSequence< _Tp, _DifferenceTp ></code> Class Template Reference	1094
5.159.1 Detailed Description	1094
5.159.2 Constructor & Destructor Documentation	1094
5.159.3 Member Function Documentation	1094
5.160 <code>__gnu_parallel::_PseudoSequenceIterator< _Tp, _DifferenceTp ></code> Class Template Reference	1095

5.160.1 Detailed Description	1095
5.161 <code>__gnu_parallel::QSBThreadLocal<_RAIter></code> Struct Template Reference	1096
5.161.1 Detailed Description	1096
5.161.2 Member Typedef Documentation	1096
5.161.3 Constructor & Destructor Documentation	1096
5.161.4 Member Data Documentation	1097
5.162 <code>__gnu_parallel::RandomNumber</code> Class Reference	1098
5.162.1 Detailed Description	1098
5.162.2 Constructor & Destructor Documentation	1098
5.162.3 Member Function Documentation	1098
5.163 <code>__gnu_parallel::RestrictedBoundedConcurrentQueue<_Tp></code> Class Template Reference	1099
5.163.1 Detailed Description	1099
5.163.2 Constructor & Destructor Documentation	1100
5.163.3 Member Function Documentation	1100
5.164 <code>__gnu_parallel::SamplingSorter<__stable, _RAIter, _StrictWeakOrdering></code> Struct Template Reference	1101
5.164.1 Detailed Description	1101
5.165 <code>__gnu_parallel::SamplingSorter<false, _RAIter, _StrictWeakOrdering></code> Struct Template Reference	1101
5.165.1 Detailed Description	1101
5.166 <code>__gnu_parallel::Settings</code> Struct Reference	1102
5.166.1 Detailed Description	1103
5.166.2 Member Function Documentation	1103
5.166.3 Member Data Documentation	1103
5.167 <code>__gnu_parallel::SplitConsistently<__exact, _RAIter, _Compare, _SortingPlacesIterator></code> Struct Template Reference	1109
5.167.1 Detailed Description	1109
5.168 <code>__gnu_parallel::SplitConsistently<false, _RAIter, _Compare, _SortingPlacesIterator></code> Struct Template Reference	1110
5.168.1 Detailed Description	1110

5.169 __gnu_parallel::_SplitConsistently< true, _RAIter, _Compare, _SortingPlacesIterator > Struct Template Reference	1110
5.169.1 Detailed Description	1110
5.170 __gnu_parallel::balanced_quicksort_tag Struct Reference	1111
5.170.1 Detailed Description	1111
5.170.2 Member Function Documentation	1111
5.171 __gnu_parallel::balanced_tag Struct Reference	1112
5.171.1 Detailed Description	1112
5.171.2 Member Function Documentation	1112
5.172 __gnu_parallel::constant_size_blocks_tag Struct Reference	1113
5.172.1 Detailed Description	1113
5.173 __gnu_parallel::default_parallel_tag Struct Reference	1114
5.173.1 Detailed Description	1114
5.173.2 Member Function Documentation	1114
5.174 __gnu_parallel::equal_split_tag Struct Reference	1115
5.174.1 Detailed Description	1115
5.175 __gnu_parallel::exact_tag Struct Reference	1116
5.175.1 Detailed Description	1116
5.175.2 Member Function Documentation	1116
5.176 __gnu_parallel::find_tag Struct Reference	1117
5.176.1 Detailed Description	1117
5.177 __gnu_parallel::growing_blocks_tag Struct Reference	1118
5.177.1 Detailed Description	1118
5.178 __gnu_parallel::multiway_mergesort_exact_tag Struct Reference	1118
5.178.1 Detailed Description	1119
5.178.2 Member Function Documentation	1119
5.179 __gnu_parallel::multiway_mergesort_sampling_tag Struct Reference	1120
5.179.1 Detailed Description	1120

5.179.2 Member Function Documentation	1120
5.180 __gnu_parallel::multiway_mergesort_tag Struct Reference	1121
5.180.1 Detailed Description	1121
5.180.2 Member Function Documentation	1122
5.181 __gnu_parallel::omp_loop_static_tag Struct Reference	1122
5.181.1 Detailed Description	1123
5.181.2 Member Function Documentation	1123
5.182 __gnu_parallel::omp_loop_tag Struct Reference	1124
5.182.1 Detailed Description	1124
5.182.2 Member Function Documentation	1124
5.183 __gnu_parallel::parallel_tag Struct Reference	1126
5.183.1 Detailed Description	1127
5.183.2 Constructor & Destructor Documentation	1127
5.183.3 Member Function Documentation	1127
5.184 __gnu_parallel::quicksort_tag Struct Reference	1128
5.184.1 Detailed Description	1128
5.184.2 Member Function Documentation	1129
5.185 __gnu_parallel::sampling_tag Struct Reference	1129
5.185.1 Detailed Description	1130
5.185.2 Member Function Documentation	1130
5.186 __gnu_parallel::sequential_tag Struct Reference	1130
5.186.1 Detailed Description	1131
5.187 __gnu_parallel::unbalanced_tag Struct Reference	1131
5.187.1 Detailed Description	1131
5.187.2 Member Function Documentation	1131
5.188 __gnu_pbds::associative_tag Struct Reference	1132
5.188.1 Detailed Description	1132

5.189 __gnu_pbds::basic_branch< Key, Mapped, Tag, Node_Update, Policy_Tl, _Alloc > Class Template Reference	1132
5.189.1 Detailed Description	1133
5.190 __gnu_pbds::basic_branch_tag Struct Reference	1134
5.190.1 Detailed Description	1134
5.191 __gnu_pbds::basic_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Resize_Policy, Store_Hash, Tag, Policy_Tl, _Alloc > Class Template Reference	1134
5.191.1 Detailed Description	1135
5.192 __gnu_pbds::basic_hash_tag Struct Reference	1136
5.192.1 Detailed Description	1136
5.193 __gnu_pbds::basic_invalidation_guarantee Struct Reference	1137
5.193.1 Detailed Description	1137
5.194 __gnu_pbds::binary_heap_tag Struct Reference	1138
5.194.1 Detailed Description	1138
5.195 __gnu_pbds::binomial_heap_tag Struct Reference	1139
5.195.1 Detailed Description	1139
5.196 __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type > Class Template Reference	1139
5.196.1 Detailed Description	1140
5.196.2 Member Enumeration Documentation	1140
5.196.3 Constructor & Destructor Documentation	1140
5.196.4 Member Function Documentation	1141
5.197 __gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc > Class Template Reference	1144
5.197.1 Detailed Description	1145
5.197.2 Constructor & Destructor Documentation	1145
5.198 __gnu_pbds::cc_hash_tag Struct Reference	1148
5.198.1 Detailed Description	1148
5.199 __gnu_pbds::container_error Struct Reference	1149
5.199.1 Detailed Description	1149

5.199.2 Member Function Documentation	1149
5.200 __gnu_pbds::container_tag Struct Reference	1150
5.200.1 Detailed Description	1150
5.201 __gnu_pbds::container_traits< Cntnr > Struct Template Reference	1150
5.201.1 Detailed Description	1151
5.201.2 Member Enumeration Documentation	1151
5.202 __gnu_pbds::container_traits_base< _Tag > Struct Template Reference	1151
5.202.1 Detailed Description	1151
5.203 __gnu_pbds::container_traits_base< binary_heap_tag > Struct Template Reference	1152
5.203.1 Detailed Description	1152
5.204 __gnu_pbds::container_traits_base< binomial_heap_tag > Struct Template Reference	1152
5.204.1 Detailed Description	1152
5.205 __gnu_pbds::container_traits_base< cc_hash_tag > Struct Template Reference	1153
5.205.1 Detailed Description	1153
5.206 __gnu_pbds::container_traits_base< gp_hash_tag > Struct Template Reference	1153
5.206.1 Detailed Description	1153
5.207 __gnu_pbds::container_traits_base< list_update_tag > Struct Template Reference	1154
5.207.1 Detailed Description	1154
5.208 __gnu_pbds::container_traits_base< ov_tree_tag > Struct Template Reference	1154
5.208.1 Detailed Description	1154
5.209 __gnu_pbds::container_traits_base< pairing_heap_tag > Struct Template Reference	1155
5.209.1 Detailed Description	1155
5.210 __gnu_pbds::container_traits_base< pat_trie_tag > Struct Template Reference	1155
5.210.1 Detailed Description	1155
5.211 __gnu_pbds::container_traits_base< rb_tree_tag > Struct Template Reference	1156
5.211.1 Detailed Description	1156
5.212 __gnu_pbds::container_traits_base< rc_binomial_heap_tag > Struct Template Reference	1156
5.212.1 Detailed Description	1156

5.213__gnu_pbds::container_traits_base< splay_tree_tag > Struct Template Reference	1157
5.213.1 Detailed Description	1157
5.214__gnu_pbds::container_traits_base< thin_heap_tag > Struct Template Reference	1157
5.214.1 Detailed Description	1157
5.215__gnu_pbds::detail::bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > Class Template Reference	1158
5.215.1 Detailed Description	1159
5.216__gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc > Class Template Reference	1159
5.216.1 Detailed Description	1160
5.216.2 Member Typedef Documentation	1160
5.216.3 Member Function Documentation	1162
5.217__gnu_pbds::detail::bin_search_tree_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > Class Template Reference	1163
5.217.1 Detailed Description	1164
5.218__gnu_pbds::detail::bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc > Class Template Reference	1165
5.218.1 Detailed Description	1165
5.218.2 Member Typedef Documentation	1166
5.218.3 Member Function Documentation	1167
5.219__gnu_pbds::detail::bin_search_tree_traits< Key, Mapped, Cmp_Fn, Node_Update, Node, _Alloc > Struct Template Reference	1168
5.219.1 Detailed Description	1168
5.219.2 Member Typedef Documentation	1169
5.220__gnu_pbds::detail::bin_search_tree_traits< Key, null_type, Cmp_Fn, Node_Update, Node, _Alloc > Struct Template Reference	1169
5.220.1 Detailed Description	1169
5.220.2 Member Typedef Documentation	1170
5.221__gnu_pbds::detail::binary_heap< Value_Type, Cmp_Fn, _Alloc > Class Template Reference	1170
5.221.1 Detailed Description	1172

5.222__gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc > Class Template Reference	1172
5.222.1 Detailed Description	1173
5.222.2 Member Typedef Documentation	1173
5.222.3 Constructor & Destructor Documentation	1174
5.222.4 Member Function Documentation	1175
5.223__gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc > Class Template Reference	1176
5.223.1 Detailed Description	1177
5.223.2 Member Typedef Documentation	1177
5.223.3 Constructor & Destructor Documentation	1178
5.223.4 Member Function Documentation	1179
5.224__gnu_pbds::detail::binomial_heap< Value_Type, Cmp_Fn, _Alloc > Class Template Reference	1179
5.224.1 Detailed Description	1181
5.225__gnu_pbds::detail::binomial_heap_base< Value_Type, Cmp_Fn, _Alloc > Class Template Reference	1182
5.225.1 Detailed Description	1184
5.226__gnu_pbds::detail::branch_policy< Node_Cltr, Node_Itr, _Alloc > Struct Template Reference	1184
5.226.1 Detailed Description	1185
5.227__gnu_pbds::detail::branch_policy< Node_Cltr, Node_Cltr, _Alloc > Struct Template Reference	1185
5.227.1 Detailed Description	1186
5.228__gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy > Class Template Reference	1186
5.228.1 Detailed Description	1188
5.228.2 Member Enumeration Documentation	1189
5.228.3 Member Function Documentation	1189
5.229__gnu_pbds::detail::cond_dealtor< Entry, _Alloc > Class Template Reference	1190
5.229.1 Detailed Description	1191
5.230__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, Tag, Policy_TI > Struct Template Reference	1191
5.230.1 Detailed Description	1191

5.231 <code>__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, binary_heap_tag, null_type ></code> Struct Template Reference	1192
5.231.1 Detailed Description	1192
5.231.2 Member Typedef Documentation	1192
5.232 <code>__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, binomial_heap_tag, null_type ></code> Struct Template Reference	1192
5.232.1 Detailed Description	1192
5.232.2 Member Typedef Documentation	1193
5.233 <code>__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, pairing_heap_tag, null_type ></code> Struct Template Reference	1193
5.233.1 Detailed Description	1193
5.233.2 Member Typedef Documentation	1193
5.234 <code>__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, rc_binomial_heap_tag, null_type ></code> Struct Template Reference	1194
5.234.1 Detailed Description	1194
5.234.2 Member Typedef Documentation	1194
5.235 <code>__gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, thin_heap_tag, null_type ></code> Struct Template Reference	1194
5.235.1 Detailed Description	1194
5.235.2 Member Typedef Documentation	1195
5.236 <code>__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, cc_hash_tag, Policy_TI ></code> Struct Template Reference	1195
5.236.1 Detailed Description	1195
5.236.2 Member Typedef Documentation	1195
5.237 <code>__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, gp_hash_tag, Policy_TI ></code> Struct Template Reference	1196
5.237.1 Detailed Description	1196
5.237.2 Member Typedef Documentation	1196
5.238 <code>__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, list_update_tag, Policy_TI ></code> Struct Template Reference	1196
5.238.1 Detailed Description	1196
5.238.2 Member Typedef Documentation	1197

5.239__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, ov_tree_tag, Policy_TI > Struct Template Reference	1197
5.239.1 Detailed Description	1197
5.239.2 Member Typedef Documentation	1197
5.240__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, pat_trie_tag, Policy_TI > Struct Template Reference	1198
5.240.1 Detailed Description	1198
5.241__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, rb_tree_tag, Policy_TI > Struct Template Reference	1198
5.241.1 Detailed Description	1198
5.241.2 Member Typedef Documentation	1198
5.242__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, splay_tree_tag, Policy_TI > Struct Template Reference	1199
5.242.1 Detailed Description	1199
5.242.2 Member Typedef Documentation	1199
5.243__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, cc_hash_tag, Policy_TI > Struct Template Reference	1199
5.243.1 Detailed Description	1199
5.243.2 Member Typedef Documentation	1200
5.244__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, gp_hash_tag, Policy_TI > Struct Template Reference	1200
5.244.1 Detailed Description	1200
5.244.2 Member Typedef Documentation	1200
5.245__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, list_update_tag, Policy_TI > Struct Template Reference	1201
5.245.1 Detailed Description	1201
5.245.2 Member Typedef Documentation	1201
5.246__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, ov_tree_tag, Policy_TI > Struct Template Reference	1201
5.246.1 Detailed Description	1201
5.246.2 Member Typedef Documentation	1202

5.247 __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, pat_trie_tag, Policy_TI > Struct Template Reference	1202
5.247.1 Detailed Description	1202
5.247.2 Member Typedef Documentation	1202
5.248 __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, rb_tree_tag, Policy_TI > Struct Template Reference	1203
5.248.1 Detailed Description	1203
5.249 __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, splay_tree_tag, Policy_TI > Struct Template Reference	1203
5.249.1 Detailed Description	1203
5.249.2 Member Typedef Documentation	1203
5.250 __gnu_pbds::detail::default_comb_hash_fn Struct Reference	1204
5.250.1 Detailed Description	1204
5.250.2 Member Typedef Documentation	1204
5.251 __gnu_pbds::detail::default_eq_fn< Key > Struct Template Reference	1204
5.251.1 Detailed Description	1204
5.251.2 Member Typedef Documentation	1205
5.252 __gnu_pbds::detail::default_hash_fn< Key > Struct Template Reference	1205
5.252.1 Detailed Description	1205
5.252.2 Member Typedef Documentation	1205
5.253 __gnu_pbds::detail::default_probe_fn< Comb_Probe_Fn > Struct Template Reference	1205
5.253.1 Detailed Description	1206
5.253.2 Member Typedef Documentation	1206
5.254 __gnu_pbds::detail::default_resize_policy< Comb_Hash_Fn > Struct Template Reference	1206
5.254.1 Detailed Description	1206
5.254.2 Member Typedef Documentation	1206
5.255 __gnu_pbds::detail::default_trie_access_traits< Key > Struct Template Reference	1207
5.255.1 Detailed Description	1207
5.256 __gnu_pbds::detail::default_trie_access_traits< std::basic_string< Char, Char_Traits, std::allocator< char > > > Struct Template Reference	1207

5.256.1 Detailed Description	1207
5.256.2 Member Typedef Documentation	1207
5.257 __gnu_pbds::detail::default_update_policy Struct Reference	1208
5.257.1 Detailed Description	1208
5.257.2 Member Typedef Documentation	1208
5.258 __gnu_pbds::detail::dumnode_const_iterator< Key, Data, _Alloc > Struct Template Reference	1208
5.258.1 Detailed Description	1208
5.259 __gnu_pbds::detail::entry_cmp< _VTp, Cmp_Fn, _Alloc, No_Throw > Struct Template Reference	1209
5.259.1 Detailed Description	1209
5.260 __gnu_pbds::detail::entry_cmp< _VTp, Cmp_Fn, _Alloc, false > Struct Template Reference	1209
5.260.1 Detailed Description	1209
5.261 __gnu_pbds::detail::entry_cmp< _VTp, Cmp_Fn, _Alloc, false >::type Struct Reference	1209
5.261.1 Detailed Description	1210
5.262 __gnu_pbds::detail::entry_cmp< _VTp, Cmp_Fn, _Alloc, true > Struct Template Reference	1210
5.262.1 Detailed Description	1210
5.262.2 Member Typedef Documentation	1210
5.263 __gnu_pbds::detail::entry_pred< _VTp, Pred, _Alloc, No_Throw > Struct Template Reference	1211
5.263.1 Detailed Description	1211
5.264 __gnu_pbds::detail::entry_pred< _VTp, Pred, _Alloc, false > Struct Template Reference	1211
5.264.1 Detailed Description	1211
5.265 __gnu_pbds::detail::entry_pred< _VTp, Pred, _Alloc, true > Struct Template Reference	1211
5.265.1 Detailed Description	1212
5.266 __gnu_pbds::detail::eq_by_less< Key, Cmp_Fn > Struct Template Reference	1212
5.266.1 Detailed Description	1212
5.267 __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy > Class Template Reference	1212
5.267.1 Detailed Description	1214
5.267.2 Member Enumeration Documentation	1215

5.267.3 Member Function Documentation	1215
5.268__gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, Store_Hash > Struct Template Reference . . .	1217
5.268.1 Detailed Description	1218
5.269__gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, false > Struct Template Reference	1218
5.269.1 Detailed Description	1218
5.270__gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, true > Struct Template Reference	1219
5.270.1 Detailed Description	1219
5.271__gnu_pbds::detail::hash_load_check_resize_trigger_size_base< Size_Type, Hold_Size > Class Template Reference	1219
5.271.1 Detailed Description	1219
5.272__gnu_pbds::detail::hash_load_check_resize_trigger_size_base< Size_Type, true > Class Template Reference	1220
5.272.1 Detailed Description	1220
5.273__gnu_pbds::detail::left_child_next_sibling_heap< Value_Type, Cmp_Fn, Node_Metadata, _Alloc > Class Template Reference	1220
5.273.1 Detailed Description	1222
5.274__gnu_pbds::detail::left_child_next_sibling_heap_const_iterator< Node, _Alloc > Class Template Reference	1222
5.274.1 Detailed Description	1223
5.274.2 Member Typedef Documentation	1223
5.274.3 Constructor & Destructor Documentation	1224
5.274.4 Member Function Documentation	1224
5.275__gnu_pbds::detail::left_child_next_sibling_heap_node< _Value, _Metadata, _Alloc > Struct Template Reference	1225
5.275.1 Detailed Description	1226
5.276__gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator< Node, _Alloc > Class Template Reference	1226
5.276.1 Detailed Description	1227
5.276.2 Member Typedef Documentation	1227
5.276.3 Constructor & Destructor Documentation	1229
5.276.4 Member Function Documentation	1229

5.277__gnu_pbds::detail::lu_counter_metadata< Size_Type > Class Template Reference	1230
5.277.1 Detailed Description	1230
5.278__gnu_pbds::detail::lu_counter_policy_base< Size_Type > Class Template Reference	1230
5.278.1 Detailed Description	1231
5.279__gnu_pbds::detail::lu_map< Key, Mapped, Eq_Fn, _Alloc, Update_Policy > Class Template Reference	1231
5.279.1 Detailed Description	1233
5.280__gnu_pbds::detail::mask_based_range_hashing< Size_Type > Class Template Reference	1233
5.280.1 Detailed Description	1234
5.281__gnu_pbds::detail::mod_based_range_hashing< Size_Type > Class Template Reference	1234
5.281.1 Detailed Description	1235
5.282__gnu_pbds::detail::no_throw_copies< Key, Mapped > Struct Template Reference	1235
5.282.1 Detailed Description	1235
5.283__gnu_pbds::detail::no_throw_copies< Key, null_type > Struct Template Reference	1236
5.283.1 Detailed Description	1236
5.284__gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > Class Template Reference	1236
5.284.1 Detailed Description	1238
5.284.2 Member Function Documentation	1238
5.285__gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::cond_dtor< Size_Type > Class Template Reference	1239
5.285.1 Detailed Description	1240
5.286__gnu_pbds::detail::ov_tree_node_const_it_< Value_Type, Metadata_Type, _Alloc > Class Template Reference	1240
5.286.1 Detailed Description	1241
5.286.2 Member Function Documentation	1241
5.287__gnu_pbds::detail::ov_tree_node_it_< Value_Type, Metadata_Type, _Alloc > Class Template Reference	1242
5.287.1 Detailed Description	1243
5.287.2 Member Function Documentation	1243
5.288__gnu_pbds::detail::pairing_heap< Value_Type, Cmp_Fn, _Alloc > Class Template Reference	1244

5.288.1 Detailed Description	1246
5.289__gnu_pbds::detail::pat_trie_base Struct Reference	1246
5.289.1 Detailed Description	1247
5.289.2 Member Enumeration Documentation	1247
5.290__gnu_pbds::detail::pat_trie_base::_Clter< Node, Leaf, Head, Inode, Is_Forward_Iterator > Class Template Reference	1247
5.290.1 Detailed Description	1249
5.291__gnu_pbds::detail::pat_trie_base::_Head< _ATraits, Metadata > Struct Template Reference	1249
5.291.1 Detailed Description	1250
5.292__gnu_pbds::detail::pat_trie_base::_Inode< _ATraits, Metadata > Struct Template Reference	1250
5.292.1 Detailed Description	1252
5.293__gnu_pbds::detail::pat_trie_base::_Inode< _ATraits, Metadata >::const_iterator Struct Reference	1252
5.293.1 Detailed Description	1253
5.294__gnu_pbds::detail::pat_trie_base::_Inode< _ATraits, Metadata >::iterator Struct Reference	1253
5.294.1 Detailed Description	1254
5.295__gnu_pbds::detail::pat_trie_base::_Iter< Node, Leaf, Head, Inode, Is_Forward_Iterator > Class Template Reference	1255
5.295.1 Detailed Description	1256
5.296__gnu_pbds::detail::pat_trie_base::_Leaf< _ATraits, Metadata > Struct Template Reference	1257
5.296.1 Detailed Description	1258
5.297__gnu_pbds::detail::pat_trie_base::_Metadata< Metadata, _Alloc > Struct Template Reference	1258
5.297.1 Detailed Description	1258
5.298__gnu_pbds::detail::pat_trie_base::_Metadata< null_type, _Alloc > Struct Template Reference	1259
5.298.1 Detailed Description	1259
5.299__gnu_pbds::detail::pat_trie_base::_Node_base< _ATraits, Metadata > Struct Template Reference	1259
5.299.1 Detailed Description	1260
5.300__gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _Clterator, Iterator, _Alloc > Class Template Reference	1260
5.300.1 Detailed Description	1261

5.300.2 Member Typedef Documentation	1262
5.300.3 Member Function Documentation	1262
5.301 <code>__gnu_pbds::detail::pat_trie_base::_Node_iter< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc ></code> Class Template Reference	1263
5.301.1 Detailed Description	1264
5.301.2 Member Typedef Documentation	1265
5.301.3 Member Function Documentation	1265
5.302 <code>__gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc ></code> Class Template Refer- ence	1266
5.302.1 Detailed Description	1268
5.302.2 Member Enumeration Documentation	1269
5.302.3 Member Function Documentation	1269
5.303 <code>__gnu_pbds::detail::probe_fn_base< _Alloc ></code> Class Template Reference	1270
5.303.1 Detailed Description	1270
5.304 <code>__gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, Store_Hash ></code> Class Template Reference	1270
5.304.1 Detailed Description	1270
5.305 <code>__gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, false ></code> Class Template Reference	1271
5.305.1 Detailed Description	1271
5.306 <code>__gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, true ></code> Class Template Reference	1271
5.306.1 Detailed Description	1272
5.307 <code>__gnu_pbds::detail::ranged_hash_fn< Key, null_type, _Alloc, Comb_Hash_Fn, false ></code> Class Template Reference	1272
5.307.1 Detailed Description	1273
5.308 <code>__gnu_pbds::detail::ranged_hash_fn< Key, null_type, _Alloc, Comb_Hash_Fn, true ></code> Class Template Reference	1273
5.308.1 Detailed Description	1273
5.309 <code>__gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, Store_↵ Hash ></code> Class Template Reference	1274
5.309.1 Detailed Description	1274

5.310 __gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, false > Class Template Reference	1274
5.310.1 Detailed Description	1275
5.311 __gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, true > Class Template Reference	1275
5.311.1 Detailed Description	1276
5.312 __gnu_pbds::detail::ranged_probe_fn< Key, null_type, _Alloc, Comb_Probe_Fn, null_type, false > Class Template Reference	1276
5.312.1 Detailed Description	1276
5.313 __gnu_pbds::detail::rb_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > Class Template Reference	1277
5.313.1 Detailed Description	1279
5.313.2 Member Function Documentation	1280
5.314 __gnu_pbds::detail::rb_tree_node_< Value_Type, Metadata, _Alloc > Struct Template Reference	1280
5.314.1 Detailed Description	1281
5.315 __gnu_pbds::detail::rc< _Node, _Alloc > Class Template Reference	1281
5.315.1 Detailed Description	1282
5.316 __gnu_pbds::detail::rc_binomial_heap< Value_Type, Cmp_Fn, _Alloc > Class Template Reference	1282
5.316.1 Detailed Description	1284
5.317 __gnu_pbds::detail::resize_policy< _Tp > Class Template Reference	1284
5.317.1 Detailed Description	1285
5.318 __gnu_pbds::detail::splay_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > Class Template Reference	1285
5.318.1 Detailed Description	1288
5.318.2 Member Function Documentation	1288
5.319 __gnu_pbds::detail::splay_tree_node_< Value_Type, Metadata, _Alloc > Struct Template Reference	1289
5.319.1 Detailed Description	1289
5.320 __gnu_pbds::detail::stored_data< _Tv, _Th > Struct Template Reference	1290
5.320.1 Detailed Description	1290
5.321 __gnu_pbds::detail::stored_data< _Tv, null_type > Struct Template Reference	1291

5.321.1 Detailed Description	1291
5.322__gnu_pbds::detail::stored_hash< _Th > Struct Template Reference	1292
5.322.1 Detailed Description	1292
5.323__gnu_pbds::detail::stored_value< _Tv > Struct Template Reference	1293
5.323.1 Detailed Description	1293
5.324__gnu_pbds::detail::synth_access_traits< Type_Traits, Set, _ATraits > Struct Template Reference	1293
5.324.1 Detailed Description	1294
5.325__gnu_pbds::detail::thin_heap< Value_Type, Cmp_Fn, _Alloc > Class Template Reference	1294
5.325.1 Detailed Description	1296
5.326__gnu_pbds::detail::tree_metadata_helper< Node_Update, _BTp > Struct Template Reference	1297
5.326.1 Detailed Description	1297
5.327__gnu_pbds::detail::tree_metadata_helper< Node_Update, false > Struct Template Reference	1297
5.327.1 Detailed Description	1297
5.328__gnu_pbds::detail::tree_metadata_helper< Node_Update, true > Struct Template Reference	1297
5.328.1 Detailed Description	1298
5.329__gnu_pbds::detail::tree_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc > Struct Template Reference	1298
5.329.1 Detailed Description	1298
5.330__gnu_pbds::detail::tree_traits< Key, Data, Cmp_Fn, Node_Update, Tag, _Alloc > Struct Template Ref- erence	1298
5.330.1 Detailed Description	1298
5.331__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc > Struct Template Reference	1299
5.331.1 Detailed Description	1299
5.331.2 Member Typedef Documentation	1299
5.332__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc > Struct Template Reference	1299
5.332.1 Detailed Description	1300
5.332.2 Member Typedef Documentation	1300

5.333__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc > Struct Template Reference	1300
5.333.1 Detailed Description	1301
5.333.2 Member Typedef Documentation	1301
5.334__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc > Struct Template Reference	1302
5.334.1 Detailed Description	1303
5.334.2 Member Typedef Documentation	1303
5.335__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc > Struct Template Reference	1303
5.335.1 Detailed Description	1304
5.335.2 Member Typedef Documentation	1304
5.336__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc > Struct Template Reference	1305
5.336.1 Detailed Description	1305
5.336.2 Member Typedef Documentation	1306
5.337__gnu_pbds::detail::trie_metadata_helper< Node_Update, _BTp > Struct Template Reference	1306
5.337.1 Detailed Description	1306
5.338__gnu_pbds::detail::trie_metadata_helper< Node_Update, false > Struct Template Reference	1306
5.338.1 Detailed Description	1306
5.339__gnu_pbds::detail::trie_metadata_helper< Node_Update, true > Struct Template Reference	1307
5.339.1 Detailed Description	1307
5.340__gnu_pbds::detail::trie_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc > Struct Template Reference	1307
5.340.1 Detailed Description	1307
5.341__gnu_pbds::detail::trie_policy_base< Node_Cltr, Node_Itr, _ATraits, _Alloc > Class Template Reference	1308
5.341.1 Detailed Description	1309
5.342__gnu_pbds::detail::trie_traits< Key, Data, _ATraits, Node_Update, Tag, _Alloc > Struct Template Reference	1309
5.342.1 Detailed Description	1309

5.343 __gnu_pbds::detail::trie_traits< Key, Mapped, _ATraits, Node_Update, pat_trie_tag, _Alloc > Struct Template Reference	1310
5.343.1 Detailed Description	1310
5.343.2 Member Typedef Documentation	1310
5.344 __gnu_pbds::detail::trie_traits< Key, null_type, _ATraits, Node_Update, pat_trie_tag, _Alloc > Struct Template Reference	1311
5.344.1 Detailed Description	1312
5.344.2 Member Typedef Documentation	1312
5.345 __gnu_pbds::detail::type_base< Key, Mapped, _Alloc, Store_Hash > Struct Template Reference	1313
5.345.1 Detailed Description	1313
5.346 __gnu_pbds::detail::type_base< Key, Mapped, _Alloc, false > Struct Template Reference	1313
5.346.1 Detailed Description	1314
5.347 __gnu_pbds::detail::type_base< Key, Mapped, _Alloc, true > Struct Template Reference	1314
5.347.1 Detailed Description	1314
5.348 __gnu_pbds::detail::type_base< Key, null_type, _Alloc, false > Struct Template Reference	1315
5.348.1 Detailed Description	1315
5.349 __gnu_pbds::detail::type_base< Key, null_type, _Alloc, true > Struct Template Reference	1315
5.349.1 Detailed Description	1316
5.350 __gnu_pbds::detail::type_dispatch< Key, Mapped, _Alloc, Store_Hash > Struct Template Reference	1316
5.350.1 Detailed Description	1316
5.351 __gnu_pbds::detail::types_traits< Key, Mapped, _Alloc, Store_Hash > Struct Template Reference	1316
5.351.1 Detailed Description	1317
5.352 __gnu_pbds::direct_mask_range_hashing< Size_Type > Class Template Reference	1317
5.352.1 Detailed Description	1318
5.352.2 Member Function Documentation	1318
5.353 __gnu_pbds::direct_mod_range_hashing< Size_Type > Class Template Reference	1319
5.353.1 Detailed Description	1319
5.353.2 Member Function Documentation	1320

5.354 <code>__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc ></code> Class Template Reference	1320
5.354.1 Detailed Description	1321
5.354.2 Constructor & Destructor Documentation	1322
5.355 <code>__gnu_pbds::gp_hash_tag</code> Struct Reference	1325
5.355.1 Detailed Description	1326
5.356 <code>__gnu_pbds::hash_exponential_size_policy< Size_Type ></code> Class Template Reference	1326
5.356.1 Detailed Description	1326
5.356.2 Constructor & Destructor Documentation	1326
5.357 <code>__gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type ></code> Class Template Reference	1327
5.357.1 Detailed Description	1328
5.357.2 Member Enumeration Documentation	1328
5.357.3 Constructor & Destructor Documentation	1328
5.357.4 Member Function Documentation	1328
5.358 <code>__gnu_pbds::hash_prime_size_policy</code> Class Reference	1329
5.358.1 Detailed Description	1330
5.358.2 Member Typedef Documentation	1330
5.358.3 Constructor & Destructor Documentation	1330
5.359 <code>__gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type ></code> Class Template Reference	1330
5.359.1 Detailed Description	1331
5.359.2 Constructor & Destructor Documentation	1331
5.359.3 Member Function Documentation	1332
5.360 <code>__gnu_pbds::insert_error</code> Struct Reference	1334
5.360.1 Detailed Description	1334
5.360.2 Member Function Documentation	1334
5.361 <code>__gnu_pbds::join_error</code> Struct Reference	1335
5.361.1 Detailed Description	1335

5.361.2 Member Function Documentation	1335
5.362__gnu_pbds::linear_probe_fn< Size_Type > Class Template Reference	1336
5.362.1 Detailed Description	1336
5.362.2 Member Function Documentation	1336
5.363__gnu_pbds::list_update< Key, Mapped, Eq_Fn, Update_Policy, _Alloc > Class Template Reference	1336
5.363.1 Detailed Description	1337
5.363.2 Constructor & Destructor Documentation	1337
5.364__gnu_pbds::list_update_tag Struct Reference	1338
5.364.1 Detailed Description	1338
5.365__gnu_pbds::lu_counter_policy< Max_Count, _Alloc > Class Template Reference	1338
5.365.1 Detailed Description	1339
5.365.2 Member Typedef Documentation	1339
5.365.3 Member Enumeration Documentation	1340
5.365.4 Member Function Documentation	1340
5.366__gnu_pbds::lu_move_to_front_policy< _Alloc > Class Template Reference	1340
5.366.1 Detailed Description	1341
5.366.2 Member Typedef Documentation	1341
5.366.3 Member Function Documentation	1341
5.367__gnu_pbds::null_node_update< _Tp1, _Tp2, _Tp3, _Tp4 > Struct Template Reference	1342
5.367.1 Detailed Description	1342
5.368__gnu_pbds::null_type Struct Reference	1342
5.368.1 Detailed Description	1343
5.369__gnu_pbds::ov_tree_tag Struct Reference	1343
5.369.1 Detailed Description	1344
5.370__gnu_pbds::pairing_heap_tag Struct Reference	1344
5.370.1 Detailed Description	1344
5.371__gnu_pbds::pat_trie_tag Struct Reference	1345
5.371.1 Detailed Description	1345

5.372__gnu_pbds::point_invalidation_guarantee Struct Reference	1346
5.372.1 Detailed Description	1346
5.373__gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc > Class Template Reference	1346
5.373.1 Detailed Description	1347
5.374__gnu_pbds::priority_queue_tag Struct Reference	1348
5.374.1 Detailed Description	1348
5.375__gnu_pbds::quadratic_probe_fn<Size_Type > Class Template Reference	1348
5.375.1 Detailed Description	1349
5.375.2 Member Function Documentation	1349
5.376__gnu_pbds::range_invalidation_guarantee Struct Reference	1349
5.376.1 Detailed Description	1350
5.377__gnu_pbds::rb_tree_tag Struct Reference	1350
5.377.1 Detailed Description	1350
5.378__gnu_pbds::rc_binomial_heap_tag Struct Reference	1351
5.378.1 Detailed Description	1351
5.379__gnu_pbds::resize_error Struct Reference	1352
5.379.1 Detailed Description	1352
5.379.2 Member Function Documentation	1352
5.380__gnu_pbds::sample_probe_fn Class Reference	1353
5.380.1 Detailed Description	1353
5.380.2 Constructor & Destructor Documentation	1353
5.380.3 Member Function Documentation	1353
5.381__gnu_pbds::sample_range_hashing Class Reference	1354
5.381.1 Detailed Description	1354
5.381.2 Member Typedef Documentation	1354
5.381.3 Constructor & Destructor Documentation	1354
5.381.4 Member Function Documentation	1355
5.382__gnu_pbds::sample_ranged_hash_fn Class Reference	1355

5.382.1 Detailed Description	1355
5.382.2 Constructor & Destructor Documentation	1356
5.382.3 Member Function Documentation	1356
5.383 __gnu_pbds::sample_ranged_probe_fn Class Reference	1356
5.383.1 Detailed Description	1357
5.384 __gnu_pbds::sample_resize_policy Class Reference	1357
5.384.1 Detailed Description	1357
5.384.2 Member Typedef Documentation	1358
5.384.3 Constructor & Destructor Documentation	1358
5.384.4 Member Function Documentation	1358
5.385 __gnu_pbds::sample_resize_trigger Class Reference	1360
5.385.1 Detailed Description	1360
5.385.2 Member Typedef Documentation	1360
5.385.3 Constructor & Destructor Documentation	1361
5.385.4 Member Function Documentation	1361
5.386 __gnu_pbds::sample_size_policy Class Reference	1362
5.386.1 Detailed Description	1363
5.386.2 Member Typedef Documentation	1363
5.386.3 Constructor & Destructor Documentation	1363
5.386.4 Member Function Documentation	1363
5.387 __gnu_pbds::sample_tree_node_update< Const_Node_Iter, Node_Iter, Cmp_Fn, _Alloc > Class Template Reference	1364
5.387.1 Detailed Description	1364
5.388 __gnu_pbds::sample_trie_access_traits Struct Reference	1364
5.388.1 Detailed Description	1365
5.388.2 Member Typedef Documentation	1365
5.388.3 Member Function Documentation	1365
5.389 __gnu_pbds::sample_trie_node_update< Node_CIter, Node_Itr, ATraits, _Alloc > Class Template Reference	1365

5.389.1 Detailed Description	1366
5.389.2 Constructor & Destructor Documentation	1366
5.389.3 Member Function Documentation	1366
5.390 __gnu_pbds::sample_update_policy Struct Reference	1366
5.390.1 Detailed Description	1367
5.390.2 Member Typedef Documentation	1367
5.390.3 Constructor & Destructor Documentation	1367
5.390.4 Member Function Documentation	1367
5.391 __gnu_pbds::sequence_tag Struct Reference	1368
5.391.1 Detailed Description	1368
5.392 __gnu_pbds::splay_tree_tag Struct Reference	1369
5.392.1 Detailed Description	1369
5.393 __gnu_pbds::string_tag Struct Reference	1370
5.393.1 Detailed Description	1370
5.394 __gnu_pbds::thin_heap_tag Struct Reference	1371
5.394.1 Detailed Description	1371
5.395 __gnu_pbds::tree< Key, Mapped, Cmp_Fn, Tag, Node_Update, _Alloc > Class Template Reference	1371
5.395.1 Detailed Description	1372
5.395.2 Member Typedef Documentation	1372
5.395.3 Constructor & Destructor Documentation	1373
5.396 __gnu_pbds::tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc > Class Template Reference	1374
5.396.1 Detailed Description	1375
5.396.2 Member Function Documentation	1375
5.397 __gnu_pbds::tree_tag Struct Reference	1376
5.397.1 Detailed Description	1377
5.398 __gnu_pbds::trie< Key, Mapped, _ATraits, Tag, Node_Update, _Alloc > Class Template Reference	1377
5.398.1 Detailed Description	1377

5.398.2 Member Typedef Documentation	1378
5.398.3 Constructor & Destructor Documentation	1378
5.399 __gnu_pbds::trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc > Class Template Reference	1379
5.399.1 Detailed Description	1381
5.399.2 Member Function Documentation	1381
5.400 __gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc > Class Template Reference	1382
5.400.1 Detailed Description	1384
5.400.2 Member Typedef Documentation	1384
5.400.3 Member Function Documentation	1384
5.401 __gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc > Struct Template Reference	1386
5.401.1 Detailed Description	1386
5.401.2 Member Typedef Documentation	1386
5.401.3 Member Function Documentation	1387
5.402 __gnu_pbds::trie_tag Struct Reference	1388
5.402.1 Detailed Description	1389
5.403 __gnu_pbds::trivial_iterator_tag Struct Reference	1389
5.403.1 Detailed Description	1389
5.404 __gnu_profile::__container_size_info Class Reference	1389
5.404.1 Detailed Description	1390
5.405 __gnu_profile::__container_size_stack_info Class Reference	1391
5.405.1 Detailed Description	1392
5.406 __gnu_profile::__hashfunc_info Class Reference	1392
5.406.1 Detailed Description	1393
5.407 __gnu_profile::__hashfunc_stack_info Class Reference	1393
5.407.1 Detailed Description	1394
5.408 __gnu_profile::__list2vector_info Class Reference	1394

5.408.1 Detailed Description	1395
5.409 __gnu_profile::__map2umap_info Class Reference	1395
5.409.1 Detailed Description	1396
5.410 __gnu_profile::__map2umap_stack_info Class Reference	1396
5.410.1 Detailed Description	1397
5.411 __gnu_profile::__object_info_base Class Reference	1397
5.411.1 Detailed Description	1398
5.412 __gnu_profile::__reentrance_guard Struct Reference	1398
5.412.1 Detailed Description	1398
5.413 __gnu_profile::__stack_hash Class Reference	1399
5.413.1 Detailed Description	1399
5.414 __gnu_profile::__trace_base< __object_info, __stack_info > Class Template Reference	1399
5.414.1 Detailed Description	1399
5.415 __gnu_profile::__trace_container_size Class Reference	1400
5.415.1 Detailed Description	1400
5.416 __gnu_profile::__trace_hash_func Class Reference	1401
5.416.1 Detailed Description	1401
5.417 __gnu_profile::__trace_hashtable_size Class Reference	1402
5.417.1 Detailed Description	1402
5.418 __gnu_profile::__trace_map2umap Class Reference	1403
5.418.1 Detailed Description	1403
5.419 __gnu_profile::__trace_vector_size Class Reference	1404
5.419.1 Detailed Description	1404
5.420 __gnu_profile::__trace_vector_to_list Class Reference	1405
5.420.1 Detailed Description	1405
5.421 __gnu_profile::__vector2list_info Class Reference	1406
5.421.1 Detailed Description	1407
5.422 __gnu_profile::__vector2list_stack_info Class Reference	1407

5.422.1 Detailed Description	1408
5.423__gnu_profile::__warning_data Struct Reference	1408
5.423.1 Detailed Description	1408
5.424const_iterator_ Class Reference	1409
5.424.1 Detailed Description	1410
5.424.2 Member Typedef Documentation	1410
5.424.3 Constructor & Destructor Documentation	1411
5.424.4 Member Function Documentation	1411
5.424.5 Member Data Documentation	1412
5.425iterator_ Class Reference	1413
5.425.1 Detailed Description	1414
5.425.2 Member Typedef Documentation	1414
5.425.3 Constructor & Destructor Documentation	1415
5.425.4 Member Function Documentation	1415
5.425.5 Member Data Documentation	1417
5.426point_const_iterator_ Class Reference	1417
5.426.1 Detailed Description	1418
5.426.2 Member Typedef Documentation	1418
5.426.3 Constructor & Destructor Documentation	1419
5.426.4 Member Function Documentation	1420
5.427point_iterator_ Class Reference	1421
5.427.1 Detailed Description	1421
5.427.2 Member Typedef Documentation	1421
5.427.3 Constructor & Destructor Documentation	1422
5.427.4 Member Function Documentation	1423
5.428std::__allocated_ptr< _Alloc > Struct Template Reference	1424
5.428.1 Detailed Description	1424
5.428.2 Constructor & Destructor Documentation	1424

5.428.3 Member Function Documentation	1425
5.429std::__atomic_base<_ITp> Struct Template Reference	1425
5.429.1 Detailed Description	1426
5.430std::__atomic_base<_PTp*> Struct Template Reference	1426
5.430.1 Detailed Description	1427
5.431std::__atomic_flag_base Struct Reference	1427
5.431.1 Detailed Description	1427
5.432std::__basic_future<_Res> Class Template Reference	1428
5.432.1 Detailed Description	1429
5.432.2 Member Typedef Documentation	1429
5.432.3 Member Function Documentation	1429
5.433std::__codecvt_abstract_base<_InternT, _ExternT, _StateT> Class Template Reference	1430
5.433.1 Detailed Description	1431
5.433.2 Member Function Documentation	1431
5.434std::__ctype_abstract_base<_CharT> Class Template Reference	1434
5.434.1 Detailed Description	1436
5.434.2 Member Typedef Documentation	1436
5.434.3 Member Function Documentation	1436
5.435std::__debug::bitset<_Nb> Class Template Reference	1448
5.435.1 Detailed Description	1450
5.436std::__debug::deque<_Tp, _Allocator> Class Template Reference	1450
5.436.1 Detailed Description	1452
5.436.2 Member Function Documentation	1453
5.436.3 Member Data Documentation	1454
5.437std::__debug::forward_list<_Tp, _Alloc> Class Template Reference	1455
5.437.1 Detailed Description	1457
5.437.2 Member Function Documentation	1457
5.437.3 Member Data Documentation	1459

5.438std::__debug::list< _Tp, _Allocator > Class Template Reference	1459
5.438.1 Detailed Description	1462
5.438.2 Member Function Documentation	1462
5.438.3 Member Data Documentation	1463
5.439std::__debug::map< _Key, _Tp, _Compare, _Allocator > Class Template Reference	1464
5.439.1 Detailed Description	1467
5.439.2 Member Function Documentation	1467
5.439.3 Member Data Documentation	1468
5.440std::__debug::multimap< _Key, _Tp, _Compare, _Allocator > Class Template Reference	1469
5.440.1 Detailed Description	1472
5.440.2 Member Function Documentation	1472
5.440.3 Member Data Documentation	1473
5.441std::__debug::multiset< _Key, _Compare, _Allocator > Class Template Reference	1474
5.441.1 Detailed Description	1477
5.441.2 Member Function Documentation	1477
5.441.3 Member Data Documentation	1478
5.442std::__debug::set< _Key, _Compare, _Allocator > Class Template Reference	1479
5.442.1 Detailed Description	1482
5.442.2 Member Function Documentation	1482
5.442.3 Member Data Documentation	1483
5.443std::__debug::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > Class Template Reference	1484
5.443.1 Detailed Description	1487
5.443.2 Member Function Documentation	1487
5.443.3 Member Data Documentation	1489
5.444std::__debug::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > Class Template Reference	1490
5.444.1 Detailed Description	1492
5.444.2 Member Function Documentation	1493
5.444.3 Member Data Documentation	1495

5.445std::__debug::unordered_multiset< _Value, _Hash, _Pred, _Alloc > Class Template Reference	1496
5.445.1 Detailed Description	1498
5.445.2 Member Function Documentation	1498
5.445.3 Member Data Documentation	1500
5.446std::__debug::unordered_set< _Value, _Hash, _Pred, _Alloc > Class Template Reference	1501
5.446.1 Detailed Description	1504
5.446.2 Member Function Documentation	1504
5.446.3 Member Data Documentation	1506
5.447std::__debug::vector< _Tp, _Allocator > Class Template Reference	1507
5.447.1 Detailed Description	1510
5.447.2 Constructor & Destructor Documentation	1510
5.447.3 Member Function Documentation	1510
5.447.4 Member Data Documentation	1512
5.448std::__detail::_BracketMatcher< _TraitsT, __icase, __collate > Struct Template Reference	1512
5.448.1 Detailed Description	1513
5.449std::__detail::_Compiler< _TraitsT > Class Template Reference	1513
5.449.1 Detailed Description	1513
5.450std::__detail::_Default_ranged_hash Struct Reference	1514
5.450.1 Detailed Description	1514
5.451std::__detail::_Equal_helper< _Key, _Value, _ExtractKey, _Equal, _HashCodeType, __cache_hash↵ code > Struct Template Reference	1514
5.451.1 Detailed Description	1514
5.452std::__detail::_Equal_helper< _Key, _Value, _ExtractKey, _Equal, _HashCodeType, false > Struct Tem- plate Reference	1514
5.452.1 Detailed Description	1514
5.453std::__detail::_Equal_helper< _Key, _Value, _ExtractKey, _Equal, _HashCodeType, true > Struct Tem- plate Reference	1515
5.453.1 Detailed Description	1515
5.454std::__detail::_Equality< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, _Unique_keys > Struct Template Reference	1515

5.454.1 Detailed Description	1516
5.455std::__detail::Equality< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, false > Struct Template Reference	1516
5.455.1 Detailed Description	1517
5.456std::__detail::Equality< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, true > Struct Template Reference	1517
5.456.1 Detailed Description	1517
5.457std::__detail::Equality_base Struct Reference	1518
5.457.1 Detailed Description	1518
5.458std::__detail::Executor< _Bilter, _Alloc, _TraitsT, __dfs_mode > Class Template Reference	1518
5.458.1 Detailed Description	1519
5.459std::__detail::Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache_hash_code > Struct Template Reference	1520
5.459.1 Detailed Description	1520
5.460std::__detail::Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Default_ranged_hash, false > Struct Template Reference	1520
5.460.1 Detailed Description	1521
5.461std::__detail::Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Default_ranged_hash, true > Struct Template Reference	1522
5.461.1 Detailed Description	1523
5.462std::__detail::Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, false > Struct Template Reference	1523
5.462.1 Detailed Description	1524
5.463std::__detail::Hash_node< _Value, _Cache_hash_code > Struct Template Reference	1524
5.463.1 Detailed Description	1524
5.464std::__detail::Hash_node< _Value, false > Struct Template Reference	1525
5.464.1 Detailed Description	1526
5.465std::__detail::Hash_node< _Value, true > Struct Template Reference	1526
5.465.1 Detailed Description	1527
5.466std::__detail::Hash_node_base Struct Reference	1527
5.466.1 Detailed Description	1528

5.467std::__detail::_Hash_node_value_base< _Value > Struct Template Reference	1528
5.467.1 Detailed Description	1529
5.468std::__detail::_Hashtable_alloc< _NodeAlloc > Struct Template Reference	1529
5.468.1 Detailed Description	1530
5.469std::__detail::_Hashtable_base< _Key, _Value, _ExtractKey, _Equal, _H1, _H2, _Hash, _Traits > Struct Template Reference	1530
5.469.1 Detailed Description	1531
5.470std::__detail::_Hashtable_ebo_helper< _Nm, _Tp, __use_ebo > Struct Template Reference	1532
5.470.1 Detailed Description	1532
5.471std::__detail::_Hashtable_ebo_helper< _Nm, _Tp, false > Struct Template Reference	1532
5.471.1 Detailed Description	1532
5.472std::__detail::_Hashtable_ebo_helper< _Nm, _Tp, true > Struct Template Reference	1532
5.472.1 Detailed Description	1533
5.473std::__detail::_Hashtable_traits< _Cache_hash_code, _Constant_iterators, _Unique_keys > Struct Template Reference	1533
5.473.1 Detailed Description	1533
5.474std::__detail::_Insert< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, _Constant_iterators, _Unique_keys > Struct Template Reference	1534
5.474.1 Detailed Description	1535
5.475std::__detail::_Insert< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, false, _Unique_keys > Struct Template Reference	1535
5.475.1 Detailed Description	1536
5.476std::__detail::_Insert< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, true, false > Struct Template Reference	1537
5.476.1 Detailed Description	1538
5.477std::__detail::_Insert< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, true, true > Struct Template Reference	1538
5.477.1 Detailed Description	1539
5.478std::__detail::_Insert_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits > Struct Template Reference	1540
5.478.1 Detailed Description	1541

5.479	<code>std::__detail::_List_node_base</code> Struct Reference	1541
5.479.1	Detailed Description	1542
5.480	<code>std::__detail::_Local_const_iterator< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __constant_↵ iterators, __cache ></code> Struct Template Reference	1542
5.480.1	Detailed Description	1543
5.481	<code>std::__detail::_Local_iterator< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __constant_iterators, __↵ cache ></code> Struct Template Reference	1543
5.481.1	Detailed Description	1544
5.482	<code>std::__detail::_Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache_hash_code ></code> Struct Template Reference	1544
5.482.1	Detailed Description	1544
5.483	<code>std::__detail::_Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, true ></code> Struct Tem- plate Reference	1545
5.483.1	Detailed Description	1546
5.484	<code>std::__detail::_Map_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _Rehash↵ Policy, _Traits, _Unique_keys ></code> Struct Template Reference	1546
5.484.1	Detailed Description	1546
5.485	<code>std::__detail::_Map_base< _Key, _Pair, _Alloc, _Select1st, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, false ></code> Struct Template Reference	1547
5.485.1	Detailed Description	1547
5.486	<code>std::__detail::_Map_base< _Key, _Pair, _Alloc, _Select1st, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, true ></code> Struct Template Reference	1547
5.486.1	Detailed Description	1547
5.487	<code>std::__detail::_Mod_range_hashing</code> Struct Reference	1548
5.487.1	Detailed Description	1548
5.488	<code>std::__detail::_Node_const_iterator< _Value, __constant_iterators, __cache ></code> Struct Template Reference	1548
5.488.1	Detailed Description	1549
5.489	<code>std::__detail::_Node_iterator< _Value, __constant_iterators, __cache ></code> Struct Template Reference	1550
5.489.1	Detailed Description	1551
5.490	<code>std::__detail::_Node_iterator_base< _Value, _Cache_hash_code ></code> Struct Template Reference	1551
5.490.1	Detailed Description	1551

5.491std::__detail::_Prime_rehash_policy Struct Reference	1552
5.491.1 Detailed Description	1552
5.492std::__detail::_Quoted_string< _String, _CharT > Struct Template Reference	1552
5.492.1 Detailed Description	1553
5.493std::__detail::_Rehash_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _Rehash↔ Policy, _Traits > Struct Template Reference	1553
5.493.1 Detailed Description	1554
5.494std::__detail::_Rehash_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _Prime↔ rehash_policy, _Traits > Struct Template Reference	1554
5.494.1 Detailed Description	1554
5.495std::__detail::_Scanner< _CharT > Class Template Reference	1555
5.495.1 Detailed Description	1556
5.495.2 Member Enumeration Documentation	1556
5.496std::__detail::_StateSeq< _TraitsT > Class Template Reference	1556
5.496.1 Detailed Description	1557
5.497std::__exception_ptr::exception_ptr Class Reference	1557
5.497.1 Detailed Description	1558
5.498std::__future_base Struct Reference	1558
5.498.1 Detailed Description	1559
5.498.2 Member Typedef Documentation	1559
5.499std::__future_base::_Result< _Res > Struct Template Reference	1559
5.499.1 Detailed Description	1560
5.500std::__future_base::_Result< _Res & > Struct Template Reference	1560
5.500.1 Detailed Description	1561
5.501std::__future_base::_Result< void > Struct Template Reference	1561
5.501.1 Detailed Description	1562
5.502std::__future_base::_Result_alloc< _Res, _Alloc > Struct Template Reference	1562
5.502.1 Detailed Description	1563
5.503std::__future_base::_Result_base Struct Reference	1563

5.503.1 Detailed Description	1564
5.504std::__is_location_invariant< _Tp > Struct Template Reference	1564
5.504.1 Detailed Description	1565
5.505std::__is_nullptr_t< _Tp > Struct Template Reference	1565
5.505.1 Detailed Description	1566
5.506std::__is_tuple_like_impl< std::pair< _T1, _T2 > > Struct Template Reference	1566
5.506.1 Detailed Description	1567
5.507std::__iterator_traits< _Iterator, typename > Struct Template Reference	1567
5.507.1 Detailed Description	1567
5.508std::__numeric_limits_base Struct Reference	1567
5.508.1 Detailed Description	1568
5.508.2 Member Data Documentation	1568
5.509std::__parallel::CRandNumber< _MustBeInt > Struct Template Reference	1572
5.509.1 Detailed Description	1572
5.510std::__profile::bitset< _Nb > Class Template Reference	1572
5.510.1 Detailed Description	1573
5.511std::__profile::deque< _Tp, _Allocator > Class Template Reference	1573
5.511.1 Detailed Description	1574
5.512std::__profile::forward_list< _Tp, _Alloc > Class Template Reference	1574
5.512.1 Detailed Description	1575
5.513std::__profile::list< _Tp, _Allocator > Class Template Reference	1575
5.513.1 Detailed Description	1577
5.514std::__profile::map< _Key, _Tp, _Compare, _Allocator > Class Template Reference	1577
5.514.1 Detailed Description	1580
5.515std::__profile::multimap< _Key, _Tp, _Compare, _Allocator > Class Template Reference	1580
5.515.1 Detailed Description	1582
5.516std::__profile::multiset< _Key, _Compare, _Allocator > Class Template Reference	1583
5.516.1 Detailed Description	1585

5.517std::__profile::set<_Key, _Compare, _Allocator > Class Template Reference	1585
5.517.1 Detailed Description	1588
5.518std::__profile::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc > Class Template Reference	1588
5.518.1 Detailed Description	1590
5.519std::__profile::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc > Class Template Reference	1590
5.519.1 Detailed Description	1592
5.520std::__profile::unordered_multiset<_Value, _Hash, _Pred, _Alloc > Class Template Reference	1592
5.520.1 Detailed Description	1593
5.521std::__profile::unordered_set<_Key, _Hash, _Pred, _Alloc > Class Template Reference	1594
5.521.1 Detailed Description	1595
5.522std::__shared_mutex_cv Class Reference	1596
5.522.1 Detailed Description	1596
5.523std::__Base_bitset<_Nw > Struct Template Reference	1596
5.523.1 Detailed Description	1597
5.523.2 Member Data Documentation	1598
5.524std::__Base_bitset<0 > Struct Template Reference	1598
5.524.1 Detailed Description	1599
5.525std::__Base_bitset<1 > Struct Template Reference	1599
5.525.1 Detailed Description	1600
5.526std::__Bind<_Signature > Struct Template Reference	1600
5.526.1 Detailed Description	1600
5.527std::__Bind_result<_Result, _Signature > Struct Template Reference	1601
5.527.1 Detailed Description	1601
5.528std::__Deque_base<_Tp, _Alloc > Class Template Reference	1601
5.528.1 Detailed Description	1602
5.528.2 Member Function Documentation	1602
5.529std::__Deque_iterator<_Tp, _Ref, _Ptr > Struct Template Reference	1603
5.529.1 Detailed Description	1604

5.529.2 Member Function Documentation	1604
5.530std::Enable_copy_move<_Copy, _CopyAssignment, _Move, _MoveAssignment, _Tag > Struct Template Reference	1605
5.530.1 Detailed Description	1605
5.531std::Enable_default_constructor<_Switch, _Tag > Struct Template Reference	1605
5.531.1 Detailed Description	1605
5.532std::Enable_destructor<_Switch, _Tag > Struct Template Reference	1606
5.532.1 Detailed Description	1606
5.533std::Enable_special_members<_Default, _Destructor, _Copy, _CopyAssignment, _Move, _MoveAssignment, _Tag > Struct Template Reference	1606
5.533.1 Detailed Description	1607
5.534std::_Function_base Class Reference	1607
5.534.1 Detailed Description	1608
5.535std::_Fwd_list_base<_Tp, _Alloc > Struct Template Reference	1608
5.535.1 Detailed Description	1609
5.536std::_Fwd_list_const_iterator<_Tp > Struct Template Reference	1609
5.536.1 Detailed Description	1610
5.537std::_Fwd_list_iterator<_Tp > Struct Template Reference	1610
5.537.1 Detailed Description	1611
5.538std::_Fwd_list_node<_Tp > Struct Template Reference	1611
5.538.1 Detailed Description	1612
5.539std::_Fwd_list_node_base Struct Reference	1612
5.539.1 Detailed Description	1613
5.540std::Hashtable<_Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits > Class Template Reference	1613
5.540.1 Detailed Description	1617
5.541std::_List_base<_Tp, _Alloc > Class Template Reference	1619
5.541.1 Detailed Description	1620
5.542std::_List_const_iterator<_Tp > Struct Template Reference	1620

5.542.1 Detailed Description	1621
5.543std::_List_iterator<_Tp> Struct Template Reference	1621
5.543.1 Detailed Description	1622
5.544std::_List_node<_Tp> Struct Template Reference	1622
5.544.1 Detailed Description	1623
5.545std::_Maybe_get_result_type<_Functor, typename> Struct Template Reference	1623
5.545.1 Detailed Description	1623
5.546std::_Maybe_unary_or_binary_function<_Res, _ArgTypes> Struct Template Reference	1624
5.546.1 Detailed Description	1624
5.547std::_Maybe_unary_or_binary_function<_Res, _T1> Struct Template Reference	1624
5.547.1 Detailed Description	1624
5.547.2 Member Typedef Documentation	1625
5.548std::_Maybe_unary_or_binary_function<_Res, _T1, _T2> Struct Template Reference	1625
5.548.1 Detailed Description	1626
5.548.2 Member Typedef Documentation	1626
5.549std::_Maybe_wrap_member_pointer<_Tp> Struct Template Reference	1626
5.549.1 Detailed Description	1627
5.550std::_Maybe_wrap_member_pointer<_Tp_Class::*> Struct Template Reference	1627
5.550.1 Detailed Description	1627
5.551std::_Mu<_Arg, _IsBindExp, _IsPlaceholder> Class Template Reference	1628
5.551.1 Detailed Description	1628
5.552std::_Mu<_Arg, false, false> Class Template Reference	1628
5.552.1 Detailed Description	1628
5.553std::_Mu<_Arg, false, true> Class Template Reference	1628
5.553.1 Detailed Description	1629
5.554std::_Mu<_Arg, true, false> Class Template Reference	1629
5.554.1 Detailed Description	1629
5.555std::_Mu<reference_wrapper<_Tp>, false, false> Class Template Reference	1629

5.555.1 Detailed Description	1630
5.556std::_Placeholder< _Num > Struct Template Reference	1630
5.556.1 Detailed Description	1630
5.557std::_Reference_wrapper_base< _Tp > Struct Template Reference	1630
5.557.1 Detailed Description	1631
5.558std::_Reference_wrapper_base_impl< _Unary, _Binary, _Tp > Struct Template Reference	1631
5.558.1 Detailed Description	1631
5.559std::_Sp_ebo_helper< _Nm, _Tp, false > Struct Template Reference	1631
5.559.1 Detailed Description	1632
5.560std::_Sp_ebo_helper< _Nm, _Tp, true > Struct Template Reference	1632
5.560.1 Detailed Description	1632
5.561std::_Temporary_buffer< _ForwardIterator, _Tp > Class Template Reference	1633
5.561.1 Detailed Description	1633
5.561.2 Constructor & Destructor Documentation	1634
5.561.3 Member Function Documentation	1634
5.562std::_Tuple_impl< _Idx, _Elements > Struct Template Reference	1635
5.562.1 Detailed Description	1635
5.563std::_Tuple_impl< _Idx, _Head, _Tail... > Struct Template Reference	1635
5.563.1 Detailed Description	1637
5.564std::_V2::condition_variable_any Class Reference	1637
5.564.1 Detailed Description	1637
5.565std::_V2::error_category Class Reference	1638
5.565.1 Detailed Description	1638
5.566std::_Vector_base< _Tp, _Alloc > Struct Template Reference	1638
5.566.1 Detailed Description	1639
5.567std::_Weak_result_type< _Functor > Struct Template Reference	1640
5.567.1 Detailed Description	1640
5.568std::_Weak_result_type_impl< _Functor > Struct Template Reference	1641

5.568.1 Detailed Description	1641
5.569std::Weak_result_type_impl< _Res(&)(_ArgTypes...)> Struct Template Reference	1641
5.569.1 Detailed Description	1641
5.570std::Weak_result_type_impl< _Res(*)(_ArgTypes...)> Struct Template Reference	1642
5.570.1 Detailed Description	1642
5.571std::Weak_result_type_impl< _Res(_ArgTypes...)> Struct Template Reference	1642
5.571.1 Detailed Description	1642
5.572std::Weak_result_type_impl< _Res(_Class::*)(_ArgTypes...) const > Struct Template Reference . . .	1642
5.572.1 Detailed Description	1643
5.573std::Weak_result_type_impl< _Res(_Class::*)(_ArgTypes...) const volatile > Struct Template Reference . .	1643
5.573.1 Detailed Description	1643
5.574std::Weak_result_type_impl< _Res(_Class::*)(_ArgTypes...) volatile > Struct Template Reference . .	1643
5.574.1 Detailed Description	1644
5.575std::Weak_result_type_impl< _Res(_Class::*)(_ArgTypes...)> Struct Template Reference	1644
5.575.1 Detailed Description	1644
5.576std::adopt_lock_t Struct Reference	1644
5.576.1 Detailed Description	1644
5.577std::allocator< _Tp > Class Template Reference	1645
5.577.1 Detailed Description	1646
5.578std::allocator< void > Class Template Reference	1647
5.578.1 Detailed Description	1647
5.579std::allocator_arg_t Struct Reference	1648
5.579.1 Detailed Description	1648
5.580std::allocator_traits< _Alloc > Struct Template Reference	1648
5.580.1 Detailed Description	1649
5.580.2 Member Typedef Documentation	1649
5.580.3 Member Function Documentation	1651
5.581std::allocator_traits< allocator< _Tp > > Struct Template Reference	1654

5.581.1 Detailed Description	1655
5.581.2 Member Typedef Documentation	1655
5.581.3 Member Function Documentation	1656
5.582std::array< _Tp, _Nm > Struct Template Reference	1659
5.582.1 Detailed Description	1660
5.583std::atomic< _Tp > Struct Template Reference	1661
5.583.1 Detailed Description	1661
5.584std::atomic< _Tp * > Struct Template Reference	1662
5.584.1 Detailed Description	1663
5.585std::atomic< bool > Struct Template Reference	1663
5.585.1 Detailed Description	1664
5.586std::atomic< char > Struct Template Reference	1664
5.586.1 Detailed Description	1665
5.587std::atomic< char16_t > Struct Template Reference	1666
5.587.1 Detailed Description	1667
5.588std::atomic< char32_t > Struct Template Reference	1667
5.588.1 Detailed Description	1668
5.589std::atomic< int > Struct Template Reference	1668
5.589.1 Detailed Description	1669
5.590std::atomic< long > Struct Template Reference	1670
5.590.1 Detailed Description	1671
5.591std::atomic< long long > Struct Template Reference	1671
5.591.1 Detailed Description	1672
5.592std::atomic< short > Struct Template Reference	1672
5.592.1 Detailed Description	1673
5.593std::atomic< signed char > Struct Template Reference	1674
5.593.1 Detailed Description	1675
5.594std::atomic< unsigned char > Struct Template Reference	1675

5.594.1 Detailed Description	1676
5.595std::atomic< unsigned int > Struct Template Reference	1676
5.595.1 Detailed Description	1677
5.596std::atomic< unsigned long > Struct Template Reference	1678
5.596.1 Detailed Description	1679
5.597std::atomic< unsigned long long > Struct Template Reference	1679
5.597.1 Detailed Description	1680
5.598std::atomic< unsigned short > Struct Template Reference	1680
5.598.1 Detailed Description	1681
5.599std::atomic< wchar_t > Struct Template Reference	1682
5.599.1 Detailed Description	1683
5.600std::atomic_flag Struct Reference	1683
5.600.1 Detailed Description	1684
5.601std::auto_ptr< _Tp > Class Template Reference	1684
5.601.1 Detailed Description	1685
5.601.2 Member Typedef Documentation	1685
5.601.3 Constructor & Destructor Documentation	1685
5.601.4 Member Function Documentation	1687
5.602std::auto_ptr_ref< _Tp1 > Struct Template Reference	1689
5.602.1 Detailed Description	1689
5.603std::back_insert_iterator< _Container > Class Template Reference	1690
5.603.1 Detailed Description	1691
5.603.2 Member Typedef Documentation	1691
5.603.3 Constructor & Destructor Documentation	1692
5.603.4 Member Function Documentation	1692
5.604std::bad_alloc Class Reference	1693
5.604.1 Detailed Description	1693
5.604.2 Member Function Documentation	1694

5.605std::bad_cast Class Reference	1694
5.605.1 Detailed Description	1694
5.605.2 Member Function Documentation	1695
5.606std::bad_exception Class Reference	1695
5.606.1 Detailed Description	1695
5.606.2 Member Function Documentation	1696
5.607std::bad_function_call Class Reference	1696
5.607.1 Detailed Description	1696
5.607.2 Member Function Documentation	1696
5.608std::bad_typeid Class Reference	1697
5.608.1 Detailed Description	1697
5.608.2 Member Function Documentation	1697
5.609std::bad_weak_ptr Class Reference	1698
5.609.1 Detailed Description	1698
5.609.2 Member Function Documentation	1698
5.610std::basic_filebuf< _CharT, _Traits > Class Template Reference	1699
5.610.1 Detailed Description	1701
5.610.2 Constructor & Destructor Documentation	1702
5.610.3 Member Function Documentation	1703
5.610.4 Member Data Documentation	1721
5.611std::basic_fstream< _CharT, _Traits > Class Template Reference	1725
5.611.1 Detailed Description	1731
5.611.2 Member Typedef Documentation	1732
5.611.3 Member Enumeration Documentation	1734
5.611.4 Constructor & Destructor Documentation	1734
5.611.5 Member Function Documentation	1735
5.611.6 Member Data Documentation	1777
5.612std::basic_ifstream< _CharT, _Traits > Class Template Reference	1784

5.612.1 Detailed Description	1789
5.612.2 Member Typedef Documentation	1790
5.612.3 Member Enumeration Documentation	1792
5.612.4 Constructor & Destructor Documentation	1792
5.612.5 Member Function Documentation	1793
5.612.6 Member Data Documentation	1826
5.613std::basic_ios< _CharT, _Traits > Class Template Reference	1832
5.613.1 Detailed Description	1836
5.613.2 Member Typedef Documentation	1836
5.613.3 Member Enumeration Documentation	1839
5.613.4 Constructor & Destructor Documentation	1839
5.613.5 Member Function Documentation	1840
5.613.6 Member Data Documentation	1854
5.614std::basic_iostream< _CharT, _Traits > Class Template Reference	1860
5.614.1 Detailed Description	1866
5.614.2 Member Typedef Documentation	1867
5.614.3 Member Enumeration Documentation	1869
5.614.4 Constructor & Destructor Documentation	1869
5.614.5 Member Function Documentation	1870
5.614.6 Member Data Documentation	1913
5.615std::basic_istream< _CharT, _Traits > Class Template Reference	1920
5.615.1 Detailed Description	1925
5.615.2 Member Typedef Documentation	1926
5.615.3 Member Enumeration Documentation	1928
5.615.4 Constructor & Destructor Documentation	1928
5.615.5 Member Function Documentation	1929
5.615.6 Member Data Documentation	1960
5.616std::basic_istream< _CharT, _Traits >::sentry Class Reference	1966

5.616.1 Detailed Description	1967
5.616.2 Member Typedef Documentation	1967
5.616.3 Constructor & Destructor Documentation	1967
5.616.4 Member Function Documentation	1968
5.617std::basic_istream< _CharT, _Traits, _Alloc > Class Template Reference	1969
5.617.1 Detailed Description	1974
5.617.2 Member Typedef Documentation	1974
5.617.3 Member Enumeration Documentation	1977
5.617.4 Constructor & Destructor Documentation	1977
5.617.5 Member Function Documentation	1978
5.617.6 Member Data Documentation	2010
5.618std::basic_ofstream< _CharT, _Traits > Class Template Reference	2017
5.618.1 Detailed Description	2021
5.618.2 Member Typedef Documentation	2022
5.618.3 Member Enumeration Documentation	2024
5.618.4 Constructor & Destructor Documentation	2024
5.618.5 Member Function Documentation	2025
5.618.6 Member Data Documentation	2051
5.619std::basic_ostream< _CharT, _Traits > Class Template Reference	2057
5.619.1 Detailed Description	2062
5.619.2 Member Typedef Documentation	2062
5.619.3 Member Enumeration Documentation	2064
5.619.4 Constructor & Destructor Documentation	2065
5.619.5 Member Function Documentation	2065
5.619.6 Member Data Documentation	2088
5.620std::basic_ostream< _CharT, _Traits >::sentry Class Reference	2094
5.620.1 Detailed Description	2095
5.620.2 Constructor & Destructor Documentation	2095

5.620.3 Member Function Documentation	2096
5.621std::basic_ostringstream< _CharT, _Traits, _Alloc > Class Template Reference	2096
5.621.1 Detailed Description	2101
5.621.2 Member Typedef Documentation	2101
5.621.3 Member Enumeration Documentation	2104
5.621.4 Constructor & Destructor Documentation	2104
5.621.5 Member Function Documentation	2105
5.621.6 Member Data Documentation	2128
5.622std::basic_regex< _Ch_type, _Rx_traits > Class Template Reference	2135
5.622.1 Detailed Description	2136
5.622.2 Constructor & Destructor Documentation	2136
5.622.3 Member Function Documentation	2139
5.623std::basic_streambuf< _CharT, _Traits > Class Template Reference	2144
5.623.1 Detailed Description	2147
5.623.2 Member Typedef Documentation	2148
5.623.3 Constructor & Destructor Documentation	2149
5.623.4 Member Function Documentation	2149
5.623.5 Member Data Documentation	2164
5.624std::basic_string< _CharT, _Traits, _Alloc > Class Template Reference	2165
5.624.1 Detailed Description	2169
5.624.2 Constructor & Destructor Documentation	2170
5.624.3 Member Function Documentation	2173
5.624.4 Member Data Documentation	2219
5.625std::basic_stringbuf< _CharT, _Traits, _Alloc > Class Template Reference	2220
5.625.1 Detailed Description	2222
5.625.2 Constructor & Destructor Documentation	2223
5.625.3 Member Function Documentation	2223
5.625.4 Member Data Documentation	2238

5.626std::basic_stringstream< _CharT, _Traits, _Alloc > Class Template Reference	2240
5.626.1 Detailed Description	2246
5.626.2 Member Typedef Documentation	2247
5.626.3 Member Enumeration Documentation	2249
5.626.4 Constructor & Destructor Documentation	2249
5.626.5 Member Function Documentation	2251
5.626.6 Member Data Documentation	2293
5.627std::bernoulli_distribution Class Reference	2299
5.627.1 Detailed Description	2300
5.627.2 Member Typedef Documentation	2300
5.627.3 Constructor & Destructor Documentation	2300
5.627.4 Member Function Documentation	2301
5.627.5 Friends And Related Function Documentation	2302
5.628std::bernoulli_distribution::param_type Struct Reference	2302
5.628.1 Detailed Description	2303
5.629std::bidirectional_iterator_tag Struct Reference	2303
5.629.1 Detailed Description	2303
5.630std::binary_function< _Arg1, _Arg2, _Result > Struct Template Reference	2304
5.630.1 Detailed Description	2304
5.630.2 Member Typedef Documentation	2304
5.631std::binary_negate< _Predicate > Class Template Reference	2305
5.631.1 Detailed Description	2306
5.631.2 Member Typedef Documentation	2306
5.632std::binder1st< _Operation > Class Template Reference	2307
5.632.1 Detailed Description	2307
5.632.2 Member Typedef Documentation	2308
5.633std::binder2nd< _Operation > Class Template Reference	2308
5.633.1 Detailed Description	2309

5.633.2 Member Typedef Documentation	2309
5.634std::binomial_distribution<_IntType> Class Template Reference	2309
5.634.1 Detailed Description	2310
5.634.2 Member Typedef Documentation	2311
5.634.3 Member Function Documentation	2311
5.634.4 Friends And Related Function Documentation	2312
5.635std::binomial_distribution<_IntType>::param_type Struct Reference	2313
5.635.1 Detailed Description	2314
5.636std::bitset<_Nb> Class Template Reference	2314
5.636.1 Detailed Description	2317
5.636.2 Constructor & Destructor Documentation	2318
5.636.3 Member Function Documentation	2319
5.637std::bitset<_Nb>::reference Class Reference	2326
5.637.1 Detailed Description	2326
5.638std::cauchy_distribution<_RealType> Class Template Reference	2327
5.638.1 Detailed Description	2327
5.638.2 Member Typedef Documentation	2328
5.638.3 Member Function Documentation	2328
5.638.4 Friends And Related Function Documentation	2329
5.639std::cauchy_distribution<_RealType>::param_type Struct Reference	2329
5.639.1 Detailed Description	2330
5.640std::char_traits<_CharT> Struct Template Reference	2330
5.640.1 Detailed Description	2331
5.641std::char_traits<__gnu_cxx::character<_Value,_Int,_St>> Struct Template Reference	2331
5.641.1 Detailed Description	2332
5.642std::char_traits<char> Struct Template Reference	2332
5.642.1 Detailed Description	2333
5.643std::char_traits<wchar_t> Struct Template Reference	2333

5.643.1 Detailed Description	2334
5.644std::chi_squared_distribution< _RealType > Class Template Reference	2334
5.644.1 Detailed Description	2335
5.644.2 Member Typedef Documentation	2336
5.644.3 Member Function Documentation	2336
5.644.4 Friends And Related Function Documentation	2337
5.645std::chi_squared_distribution< _RealType >::param_type Struct Reference	2338
5.645.1 Detailed Description	2338
5.646std::chrono::_V2::steady_clock Struct Reference	2339
5.646.1 Detailed Description	2339
5.647std::chrono::_V2::system_clock Struct Reference	2339
5.647.1 Detailed Description	2340
5.648std::chrono::duration< _Rep, _Period > Struct Template Reference	2340
5.648.1 Detailed Description	2341
5.649std::chrono::duration_values< _Rep > Struct Template Reference	2341
5.649.1 Detailed Description	2341
5.650std::chrono::time_point< _Clock, _Dur > Struct Template Reference	2341
5.650.1 Detailed Description	2342
5.651std::chrono::treat_as_floating_point< _Rep > Struct Template Reference	2342
5.651.1 Detailed Description	2343
5.652std::codecvt< _InternT, _ExternT, _StateT > Class Template Reference	2343
5.652.1 Detailed Description	2345
5.652.2 Member Function Documentation	2345
5.653std::codecvt< _InternT, _ExternT, encoding_state > Class Template Reference	2348
5.653.1 Detailed Description	2349
5.653.2 Member Function Documentation	2349
5.654std::codecvt< char, char, mbstate_t > Class Template Reference	2352
5.654.1 Detailed Description	2353

5.654.2 Member Function Documentation	2354
5.655std::codecvt< char16_t, char, mbstate_t > Class Template Reference	2357
5.655.1 Detailed Description	2358
5.655.2 Member Function Documentation	2358
5.656std::codecvt< char32_t, char, mbstate_t > Class Template Reference	2361
5.656.1 Detailed Description	2362
5.656.2 Member Function Documentation	2362
5.657std::codecvt< wchar_t, char, mbstate_t > Class Template Reference	2365
5.657.1 Detailed Description	2366
5.657.2 Member Function Documentation	2367
5.658std::codecvt_base Class Reference	2370
5.658.1 Detailed Description	2370
5.659std::codecvt_byname< _InternT, _ExternT, _StateT > Class Template Reference	2371
5.659.1 Detailed Description	2372
5.659.2 Member Function Documentation	2373
5.660std::collate< _CharT > Class Template Reference	2375
5.660.1 Detailed Description	2377
5.660.2 Member Typedef Documentation	2377
5.660.3 Constructor & Destructor Documentation	2377
5.660.4 Member Function Documentation	2378
5.660.5 Member Data Documentation	2381
5.661std::collate_byname< _CharT > Class Template Reference	2381
5.661.1 Detailed Description	2383
5.661.2 Member Typedef Documentation	2383
5.661.3 Member Function Documentation	2383
5.661.4 Member Data Documentation	2386
5.662std::complex< _Tp > Struct Template Reference	2386
5.662.1 Detailed Description	2387

5.662.2 Member Typedef Documentation	2387
5.662.3 Constructor & Destructor Documentation	2388
5.662.4 Member Function Documentation	2388
5.663std::complex< double > Struct Template Reference	2388
5.663.1 Detailed Description	2389
5.664std::complex< float > Struct Template Reference	2389
5.664.1 Detailed Description	2390
5.665std::complex< long double > Struct Template Reference	2390
5.665.1 Detailed Description	2391
5.666std::condition_variable Class Reference	2391
5.666.1 Detailed Description	2392
5.667std::const_mem_fun1_ref_t< _Ret, _Tp, _Arg > Class Template Reference	2392
5.667.1 Detailed Description	2393
5.667.2 Member Typedef Documentation	2393
5.668std::const_mem_fun1_t< _Ret, _Tp, _Arg > Class Template Reference	2394
5.668.1 Detailed Description	2394
5.668.2 Member Typedef Documentation	2394
5.669std::const_mem_fun_ref_t< _Ret, _Tp > Class Template Reference	2395
5.669.1 Detailed Description	2396
5.669.2 Member Typedef Documentation	2396
5.670std::const_mem_fun_t< _Ret, _Tp > Class Template Reference	2396
5.670.1 Detailed Description	2397
5.670.2 Member Typedef Documentation	2397
5.671std::ctype< _CharT > Class Template Reference	2398
5.671.1 Detailed Description	2400
5.671.2 Member Function Documentation	2400
5.671.3 Member Data Documentation	2412
5.672std::ctype< char > Class Template Reference	2412

5.672.1 Detailed Description	2414
5.672.2 Member Typedef Documentation	2414
5.672.3 Constructor & Destructor Documentation	2414
5.672.4 Member Function Documentation	2415
5.672.5 Member Data Documentation	2425
5.673std::ctype< wchar_t > Class Template Reference	2426
5.673.1 Detailed Description	2428
5.673.2 Member Typedef Documentation	2428
5.673.3 Constructor & Destructor Documentation	2428
5.673.4 Member Function Documentation	2429
5.673.5 Member Data Documentation	2441
5.674std::ctype_base Struct Reference	2441
5.674.1 Detailed Description	2442
5.675std::ctype_byname< _CharT > Class Template Reference	2443
5.675.1 Detailed Description	2444
5.675.2 Member Function Documentation	2445
5.675.3 Member Data Documentation	2456
5.676std::ctype_byname< char > Class Template Reference	2457
5.676.1 Detailed Description	2459
5.676.2 Member Typedef Documentation	2459
5.676.3 Member Function Documentation	2459
5.676.4 Member Data Documentation	2469
5.677std::decimal::decimal128 Class Reference	2470
5.677.1 Detailed Description	2471
5.677.2 Constructor & Destructor Documentation	2471
5.678std::decimal::decimal32 Class Reference	2472
5.678.1 Detailed Description	2473
5.678.2 Constructor & Destructor Documentation	2473

5.679	std::decimal::decimal64 Class Reference	2473
5.679.1	Detailed Description	2475
5.679.2	Constructor & Destructor Documentation	2475
5.680	std::default_delete< _Tp > Struct Template Reference	2475
5.680.1	Detailed Description	2475
5.680.2	Constructor & Destructor Documentation	2476
5.680.3	Member Function Documentation	2476
5.681	std::default_delete< _Tp[]> Struct Template Reference	2476
5.681.1	Detailed Description	2476
5.681.2	Constructor & Destructor Documentation	2477
5.681.3	Member Function Documentation	2477
5.682	std::defer_lock_t Struct Reference	2477
5.682.1	Detailed Description	2477
5.683	std::deque< _Tp, _Alloc > Class Template Reference	2478
5.683.1	Detailed Description	2482
5.683.2	Constructor & Destructor Documentation	2483
5.683.3	Member Function Documentation	2486
5.684	std::discard_block_engine< _RandomNumberEngine, __p, __r > Class Template Reference	2503
5.684.1	Detailed Description	2504
5.684.2	Member Typedef Documentation	2505
5.684.3	Constructor & Destructor Documentation	2505
5.684.4	Member Function Documentation	2506
5.684.5	Friends And Related Function Documentation	2508
5.685	std::discrete_distribution< _IntType > Class Template Reference	2509
5.685.1	Detailed Description	2510
5.685.2	Member Typedef Documentation	2510
5.685.3	Member Function Documentation	2511
5.685.4	Friends And Related Function Documentation	2512

5.686std::discrete_distribution< _IntType >::param_type Struct Reference	2513
5.686.1 Detailed Description	2513
5.687std::divides< _Tp > Struct Template Reference	2514
5.687.1 Detailed Description	2514
5.687.2 Member Typedef Documentation	2514
5.688std::divides< void > Struct Template Reference	2515
5.688.1 Detailed Description	2515
5.689std::domain_error Class Reference	2516
5.689.1 Detailed Description	2516
5.689.2 Member Function Documentation	2516
5.690std::enable_shared_from_this< _Tp > Class Template Reference	2517
5.690.1 Detailed Description	2517
5.691std::equal_to< _Tp > Struct Template Reference	2517
5.691.1 Detailed Description	2518
5.691.2 Member Typedef Documentation	2518
5.692std::equal_to< void > Struct Template Reference	2519
5.692.1 Detailed Description	2519
5.693std::error_code Struct Reference	2519
5.693.1 Detailed Description	2520
5.694std::error_condition Struct Reference	2520
5.694.1 Detailed Description	2520
5.695std::exception Class Reference	2521
5.695.1 Detailed Description	2521
5.695.2 Member Function Documentation	2522
5.696std::experimental::filesystem::v1::path Class Reference	2522
5.696.1 Detailed Description	2524
5.697std::experimental::filesystem::v1::path::iterator Class Reference	2524
5.697.1 Detailed Description	2525

5.698	std::experimental::fundamentals_v1::_Has_addressof<_Tp> Struct Template Reference	2525
5.698.1	Detailed Description	2525
5.699	std::experimental::fundamentals_v1::_Optional_base<_Tp, _ShouldProvideDestructor> Class Template Reference	2526
5.699.1	Detailed Description	2526
5.700	std::experimental::fundamentals_v1::_Optional_base<_Tp, false> Class Template Reference	2527
5.700.1	Detailed Description	2527
5.701	std::experimental::fundamentals_v1::any Class Reference	2528
5.701.1	Detailed Description	2528
5.701.2	Constructor & Destructor Documentation	2528
5.701.3	Member Function Documentation	2529
5.702	std::experimental::fundamentals_v1::bad_any_cast Class Reference	2531
5.702.1	Detailed Description	2531
5.702.2	Member Function Documentation	2531
5.703	std::experimental::fundamentals_v1::bad_optional_access Class Reference	2532
5.703.1	Detailed Description	2532
5.703.2	Member Function Documentation	2532
5.704	std::experimental::fundamentals_v1::basic_string_view<_CharT, _Traits> Class Template Reference	2533
5.704.1	Detailed Description	2534
5.705	std::experimental::fundamentals_v1::in_place_t Struct Reference	2535
5.705.1	Detailed Description	2535
5.706	std::experimental::fundamentals_v1::nullopt_t Struct Reference	2535
5.706.1	Detailed Description	2535
5.707	std::experimental::fundamentals_v1::optional<_Tp> Class Template Reference	2536
5.707.1	Detailed Description	2538
5.708	std::experimental::fundamentals_v2::_Not_fn<_Fn> Class Template Reference	2538
5.708.1	Detailed Description	2538
5.709	std::experimental::fundamentals_v2::ostream_joiner<_DelimT, _CharT, _Traits> Class Template Reference	2539

5.709.1 Detailed Description	2539
5.710std::experimental::fundamentals_v2::owner_less< shared_ptr< _Tp > > Struct Template Reference	2539
5.710.1 Detailed Description	2540
5.710.2 Member Typedef Documentation	2540
5.711std::experimental::fundamentals_v2::owner_less< weak_ptr< _Tp > > Struct Template Reference	2541
5.711.1 Detailed Description	2541
5.711.2 Member Typedef Documentation	2541
5.712std::experimental::fundamentals_v2::propagate_const< _Tp > Class Template Reference	2542
5.712.1 Detailed Description	2543
5.713std::exponential_distribution< _RealType > Class Template Reference	2543
5.713.1 Detailed Description	2544
5.713.2 Member Typedef Documentation	2544
5.713.3 Constructor & Destructor Documentation	2544
5.713.4 Member Function Documentation	2545
5.713.5 Friends And Related Function Documentation	2546
5.714std::exponential_distribution< _RealType >::param_type Struct Reference	2546
5.714.1 Detailed Description	2547
5.715std::extreme_value_distribution< _RealType > Class Template Reference	2547
5.715.1 Detailed Description	2548
5.715.2 Member Typedef Documentation	2548
5.715.3 Member Function Documentation	2548
5.715.4 Friends And Related Function Documentation	2550
5.716std::extreme_value_distribution< _RealType >::param_type Struct Reference	2550
5.716.1 Detailed Description	2550
5.717std::fisher_f_distribution< _RealType > Class Template Reference	2551
5.717.1 Detailed Description	2552
5.717.2 Member Typedef Documentation	2552
5.717.3 Member Function Documentation	2552

5.717.4 Friends And Related Function Documentation	2553
5.718std::fisher_f_distribution<_RealType>::param_type Struct Reference	2554
5.718.1 Detailed Description	2554
5.719std::forward_iterator_tag Struct Reference	2555
5.719.1 Detailed Description	2555
5.720std::forward_list<_Tp, _Alloc> Class Template Reference	2556
5.720.1 Detailed Description	2558
5.720.2 Constructor & Destructor Documentation	2559
5.720.3 Member Function Documentation	2562
5.721std::fpos<_StateT> Class Template Reference	2576
5.721.1 Detailed Description	2577
5.721.2 Constructor & Destructor Documentation	2577
5.721.3 Member Function Documentation	2577
5.722std::front_insert_iterator<_Container> Class Template Reference	2578
5.722.1 Detailed Description	2579
5.722.2 Member Typedef Documentation	2579
5.722.3 Constructor & Destructor Documentation	2580
5.722.4 Member Function Documentation	2580
5.723std::function<_Res(_ArgTypes...)> Class Template Reference	2582
5.723.1 Detailed Description	2583
5.723.2 Constructor & Destructor Documentation	2583
5.723.3 Member Function Documentation	2585
5.724std::future<_Res> Class Template Reference	2588
5.724.1 Detailed Description	2590
5.724.2 Member Typedef Documentation	2590
5.724.3 Constructor & Destructor Documentation	2590
5.724.4 Member Function Documentation	2590
5.725std::future<_Res &> Class Template Reference	2591

5.725.1 Detailed Description	2592
5.725.2 Member Typedef Documentation	2592
5.725.3 Constructor & Destructor Documentation	2593
5.725.4 Member Function Documentation	2593
5.726std::future< void > Class Template Reference	2593
5.726.1 Detailed Description	2595
5.726.2 Member Typedef Documentation	2595
5.726.3 Constructor & Destructor Documentation	2595
5.726.4 Member Function Documentation	2595
5.727std::future_error Class Reference	2596
5.727.1 Detailed Description	2596
5.727.2 Member Function Documentation	2596
5.728std::gamma_distribution< _RealType > Class Template Reference	2597
5.728.1 Detailed Description	2598
5.728.2 Member Typedef Documentation	2598
5.728.3 Constructor & Destructor Documentation	2598
5.728.4 Member Function Documentation	2598
5.728.5 Friends And Related Function Documentation	2600
5.729std::gamma_distribution< _RealType >::param_type Struct Reference	2601
5.729.1 Detailed Description	2601
5.730std::geometric_distribution< _IntType > Class Template Reference	2602
5.730.1 Detailed Description	2602
5.730.2 Member Typedef Documentation	2603
5.730.3 Member Function Documentation	2603
5.730.4 Friends And Related Function Documentation	2604
5.731std::geometric_distribution< _IntType >::param_type Struct Reference	2604
5.731.1 Detailed Description	2605
5.732std::greater< _Tp > Struct Template Reference	2605

5.732.1 Detailed Description	2605
5.732.2 Member Typedef Documentation	2606
5.733std::greater< void > Struct Template Reference	2606
5.733.1 Detailed Description	2606
5.734std::greater_equal< _Tp > Struct Template Reference	2607
5.734.1 Detailed Description	2607
5.734.2 Member Typedef Documentation	2607
5.735std::greater_equal< void > Struct Template Reference	2608
5.735.1 Detailed Description	2608
5.736std::gslice Class Reference	2609
5.736.1 Detailed Description	2609
5.737std::gslice_array< _Tp > Class Template Reference	2609
5.737.1 Detailed Description	2610
5.738std::hash< _Tp > Struct Template Reference	2611
5.738.1 Detailed Description	2611
5.739std::hash< __debug::bitset< _Nb > > Struct Template Reference	2611
5.739.1 Detailed Description	2612
5.740std::hash< __debug::vector< bool, _Alloc > > Struct Template Reference	2612
5.740.1 Detailed Description	2612
5.741std::hash< __gnu_cxx::__u16vstring > Struct Template Reference	2612
5.741.1 Detailed Description	2613
5.742std::hash< __gnu_cxx::__u32vstring > Struct Template Reference	2613
5.742.1 Detailed Description	2613
5.743std::hash< __gnu_cxx::__vstring > Struct Template Reference	2614
5.743.1 Detailed Description	2614
5.744std::hash< __gnu_cxx::__wvstring > Struct Template Reference	2614
5.744.1 Detailed Description	2615
5.745std::hash< __gnu_cxx::throw_value_limit > Struct Template Reference	2615

5.745.1 Detailed Description	2616
5.745.2 Member Typedef Documentation	2616
5.746std::hash< __gnu_cxx::throw_value_random > Struct Template Reference	2616
5.746.1 Detailed Description	2617
5.746.2 Member Typedef Documentation	2617
5.747std::hash< __profile::bitset< _Nb > > Struct Template Reference	2617
5.747.1 Detailed Description	2618
5.748std::hash< __profile::vector< bool, _Alloc > > Struct Template Reference	2618
5.748.1 Detailed Description	2618
5.749std::hash< __shared_ptr< _Tp, _Lp > > Struct Template Reference	2619
5.749.1 Detailed Description	2619
5.750std::hash< _Tp * > Struct Template Reference	2619
5.750.1 Detailed Description	2620
5.751std::hash< bool > Struct Template Reference	2620
5.751.1 Detailed Description	2620
5.752std::hash< char > Struct Template Reference	2620
5.752.1 Detailed Description	2621
5.753std::hash< char16_t > Struct Template Reference	2621
5.753.1 Detailed Description	2621
5.754std::hash< char32_t > Struct Template Reference	2622
5.754.1 Detailed Description	2622
5.755std::hash< double > Struct Template Reference	2622
5.755.1 Detailed Description	2623
5.756std::hash< error_code > Struct Template Reference	2623
5.756.1 Detailed Description	2623
5.757std::hash< experimental::shared_ptr< _Tp > > Struct Template Reference	2623
5.757.1 Detailed Description	2624
5.758std::hash< float > Struct Template Reference	2624

5.758.1 Detailed Description	2624
5.759std::hash< int > Struct Template Reference	2625
5.759.1 Detailed Description	2625
5.760std::hash< long > Struct Template Reference	2625
5.760.1 Detailed Description	2626
5.761std::hash< long double > Struct Template Reference	2626
5.761.1 Detailed Description	2626
5.762std::hash< long long > Struct Template Reference	2626
5.762.1 Detailed Description	2627
5.763std::hash< shared_ptr< _Tp > > Struct Template Reference	2627
5.763.1 Detailed Description	2627
5.764std::hash< short > Struct Template Reference	2628
5.764.1 Detailed Description	2628
5.765std::hash< signed char > Struct Template Reference	2628
5.765.1 Detailed Description	2629
5.766std::hash< string > Struct Template Reference	2629
5.766.1 Detailed Description	2629
5.767std::hash< thread::id > Struct Template Reference	2629
5.767.1 Detailed Description	2630
5.768std::hash< type_index > Struct Template Reference	2630
5.768.1 Detailed Description	2630
5.769std::hash< u16string > Struct Template Reference	2631
5.769.1 Detailed Description	2631
5.770std::hash< u32string > Struct Template Reference	2631
5.770.1 Detailed Description	2632
5.771std::hash< unique_ptr< _Tp, _Dp > > Struct Template Reference	2632
5.771.1 Detailed Description	2632
5.772std::hash< unsigned char > Struct Template Reference	2632

5.772.1 Detailed Description	2633
5.773std::hash< unsigned int > Struct Template Reference	2633
5.773.1 Detailed Description	2633
5.774std::hash< unsigned long > Struct Template Reference	2634
5.774.1 Detailed Description	2634
5.775std::hash< unsigned long long > Struct Template Reference	2634
5.775.1 Detailed Description	2635
5.776std::hash< unsigned short > Struct Template Reference	2635
5.776.1 Detailed Description	2635
5.777std::hash< wchar_t > Struct Template Reference	2635
5.777.1 Detailed Description	2636
5.778std::hash< wstring > Struct Template Reference	2636
5.778.1 Detailed Description	2636
5.779std::hash<::bitset< _Nb > > Struct Template Reference	2637
5.779.1 Detailed Description	2637
5.780std::hash<::vector< bool, _Alloc > > Struct Template Reference	2637
5.780.1 Detailed Description	2638
5.781std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > Class Template Reference	2638
5.781.1 Detailed Description	2639
5.781.2 Member Typedef Documentation	2639
5.781.3 Constructor & Destructor Documentation	2639
5.781.4 Member Function Documentation	2640
5.781.5 Friends And Related Function Documentation	2642
5.782std::indirect_array< _Tp > Class Template Reference	2643
5.782.1 Detailed Description	2644
5.783std::initializer_list< _E > Class Template Reference	2645
5.783.1 Detailed Description	2645
5.784std::input_iterator_tag Struct Reference	2646

5.784.1 Detailed Description	2646
5.785std::insert_iterator< _Container > Class Template Reference	2647
5.785.1 Detailed Description	2648
5.785.2 Member Typedef Documentation	2648
5.785.3 Constructor & Destructor Documentation	2649
5.785.4 Member Function Documentation	2649
5.786std::integer_sequence< _Tp, _Idx > Struct Template Reference	2650
5.786.1 Detailed Description	2650
5.787std::integral_constant< _Tp, __v > Struct Template Reference	2651
5.787.1 Detailed Description	2652
5.788std::invalid_argument Class Reference	2652
5.788.1 Detailed Description	2653
5.788.2 Member Function Documentation	2653
5.789std::ios_base Class Reference	2653
5.789.1 Detailed Description	2656
5.789.2 Member Typedef Documentation	2656
5.789.3 Member Enumeration Documentation	2658
5.789.4 Constructor & Destructor Documentation	2659
5.789.5 Member Function Documentation	2659
5.789.6 Member Data Documentation	2665
5.790std::ios_base::failure Class Reference	2671
5.790.1 Detailed Description	2672
5.790.2 Member Function Documentation	2672
5.791std::is_abstract< _Tp > Struct Template Reference	2672
5.791.1 Detailed Description	2673
5.792std::is_arithmetic< _Tp > Struct Template Reference	2673
5.792.1 Detailed Description	2673
5.793std::is_array< typename > Struct Template Reference	2674

5.793.1 Detailed Description	2674
5.794std::is_bind_expression< _Tp > Struct Template Reference	2675
5.794.1 Detailed Description	2675
5.795std::is_bind_expression< _Bind< _Signature > > Struct Template Reference	2676
5.795.1 Detailed Description	2676
5.796std::is_bind_expression< _Bind_result< _Result, _Signature > > Struct Template Reference	2677
5.796.1 Detailed Description	2677
5.797std::is_bind_expression< const _Bind< _Signature > > Struct Template Reference	2678
5.797.1 Detailed Description	2678
5.798std::is_bind_expression< const _Bind_result< _Result, _Signature > > Struct Template Reference	2679
5.798.1 Detailed Description	2679
5.799std::is_bind_expression< const volatile _Bind< _Signature > > Struct Template Reference	2680
5.799.1 Detailed Description	2680
5.800std::is_bind_expression< const volatile _Bind_result< _Result, _Signature > > Struct Template Reference	2681
5.800.1 Detailed Description	2681
5.801std::is_bind_expression< volatile _Bind< _Signature > > Struct Template Reference	2682
5.801.1 Detailed Description	2682
5.802std::is_bind_expression< volatile _Bind_result< _Result, _Signature > > Struct Template Reference	2683
5.802.1 Detailed Description	2683
5.803std::is_class< _Tp > Struct Template Reference	2684
5.803.1 Detailed Description	2684
5.804std::is_compound< _Tp > Struct Template Reference	2685
5.804.1 Detailed Description	2685
5.805std::is_const< typename > Struct Template Reference	2686
5.805.1 Detailed Description	2686
5.806std::is_empty< _Tp > Struct Template Reference	2687
5.806.1 Detailed Description	2687

5.807std::is_enum< _Tp > Struct Template Reference	2688
5.807.1 Detailed Description	2688
5.808std::is_error_code_enum< _Tp > Struct Template Reference	2689
5.808.1 Detailed Description	2689
5.809std::is_error_code_enum< future_errc > Struct Template Reference	2690
5.809.1 Detailed Description	2690
5.810std::is_error_condition_enum< _Tp > Struct Template Reference	2691
5.810.1 Detailed Description	2691
5.811std::is_final< _Tp > Struct Template Reference	2692
5.811.1 Detailed Description	2692
5.812std::is_floating_point< _Tp > Struct Template Reference	2693
5.812.1 Detailed Description	2693
5.813std::is_function< typename > Struct Template Reference	2693
5.813.1 Detailed Description	2694
5.814std::is_fundamental< _Tp > Struct Template Reference	2694
5.814.1 Detailed Description	2694
5.815std::is_integral< _Tp > Struct Template Reference	2694
5.815.1 Detailed Description	2695
5.816std::is_literal_type< _Tp > Struct Template Reference	2695
5.816.1 Detailed Description	2696
5.817std::is_lvalue_reference< typename > Struct Template Reference	2696
5.817.1 Detailed Description	2697
5.818std::is_member_function_pointer< _Tp > Struct Template Reference	2697
5.818.1 Detailed Description	2698
5.819std::is_member_object_pointer< _Tp > Struct Template Reference	2698
5.819.1 Detailed Description	2698
5.820std::is_member_pointer< _Tp > Struct Template Reference	2699
5.820.1 Detailed Description	2699

5.821std::is_null_pointer<_Tp> Struct Template Reference	2700
5.821.1 Detailed Description	2700
5.822std::is_object<_Tp> Struct Template Reference	2700
5.822.1 Detailed Description	2701
5.823std::is_placeholder<_Tp> Struct Template Reference	2701
5.823.1 Detailed Description	2702
5.824std::is_placeholder<_Placeholder<_Num>> Struct Template Reference	2702
5.824.1 Detailed Description	2703
5.825std::is_pod<_Tp> Struct Template Reference	2703
5.825.1 Detailed Description	2704
5.826std::is_pointer<_Tp> Struct Template Reference	2704
5.826.1 Detailed Description	2704
5.827std::is_polymorphic<_Tp> Struct Template Reference	2705
5.827.1 Detailed Description	2705
5.828std::is_reference<_Tp> Struct Template Reference	2706
5.828.1 Detailed Description	2706
5.829std::is_rvalue_reference<typename> Struct Template Reference	2706
5.829.1 Detailed Description	2707
5.830std::is_scalar<_Tp> Struct Template Reference	2707
5.830.1 Detailed Description	2707
5.831std::is_standard_layout<_Tp> Struct Template Reference	2708
5.831.1 Detailed Description	2708
5.832std::is_trivial<_Tp> Struct Template Reference	2709
5.832.1 Detailed Description	2709
5.833std::is_union<_Tp> Struct Template Reference	2710
5.833.1 Detailed Description	2710
5.834std::is_void<_Tp> Struct Template Reference	2711
5.834.1 Detailed Description	2711

5.835std::is_volatile< typename > Struct Template Reference	2712
5.835.1 Detailed Description	2712
5.836std::istream_iterator< _Tp, _CharT, _Traits, _Dist > Class Template Reference	2713
5.836.1 Detailed Description	2713
5.836.2 Member Typedef Documentation	2714
5.836.3 Constructor & Destructor Documentation	2714
5.837std::istreambuf_iterator< _CharT, _Traits > Class Template Reference	2715
5.837.1 Detailed Description	2716
5.837.2 Member Typedef Documentation	2716
5.837.3 Constructor & Destructor Documentation	2718
5.837.4 Member Function Documentation	2718
5.838std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference > Struct Template Reference	2719
5.838.1 Detailed Description	2719
5.838.2 Member Typedef Documentation	2720
5.839std::iterator_traits< _Tp * > Struct Template Reference	2720
5.839.1 Detailed Description	2721
5.840std::iterator_traits< const _Tp * > Struct Template Reference	2721
5.840.1 Detailed Description	2721
5.841std::length_error Class Reference	2722
5.841.1 Detailed Description	2722
5.841.2 Member Function Documentation	2722
5.842std::less< _Tp > Struct Template Reference	2723
5.842.1 Detailed Description	2723
5.842.2 Member Typedef Documentation	2723
5.843std::less< void > Struct Template Reference	2724
5.843.1 Detailed Description	2724
5.844std::less_equal< _Tp > Struct Template Reference	2725
5.844.1 Detailed Description	2725

5.844.2 Member Typedef Documentation	2725
5.845std::less_equal< void > Struct Template Reference	2726
5.845.1 Detailed Description	2726
5.846std::linear_congruential_engine< _UIntType, __a, __c, __m > Class Template Reference	2726
5.846.1 Detailed Description	2727
5.846.2 Member Typedef Documentation	2728
5.846.3 Constructor & Destructor Documentation	2728
5.846.4 Member Function Documentation	2728
5.846.5 Friends And Related Function Documentation	2730
5.846.6 Member Data Documentation	2731
5.847std::list< _Tp, _Alloc > Class Template Reference	2732
5.847.1 Detailed Description	2735
5.847.2 Constructor & Destructor Documentation	2736
5.847.3 Member Function Documentation	2738
5.848std::locale Class Reference	2754
5.848.1 Detailed Description	2756
5.848.2 Member Typedef Documentation	2756
5.848.3 Constructor & Destructor Documentation	2756
5.848.4 Member Function Documentation	2760
5.848.5 Friends And Related Function Documentation	2763
5.848.6 Member Data Documentation	2764
5.849std::locale::facet Class Reference	2766
5.849.1 Detailed Description	2767
5.849.2 Constructor & Destructor Documentation	2767
5.850std::locale::id Class Reference	2768
5.850.1 Detailed Description	2768
5.850.2 Constructor & Destructor Documentation	2768
5.850.3 Friends And Related Function Documentation	2768

5.851std::lock_guard< _Mutex > Class Template Reference	2770
5.851.1 Detailed Description	2770
5.852std::logic_error Class Reference	2770
5.852.1 Detailed Description	2771
5.852.2 Constructor & Destructor Documentation	2771
5.852.3 Member Function Documentation	2771
5.853std::logical_and< _Tp > Struct Template Reference	2771
5.853.1 Detailed Description	2772
5.853.2 Member Typedef Documentation	2772
5.854std::logical_and< void > Struct Template Reference	2773
5.854.1 Detailed Description	2773
5.855std::logical_not< _Tp > Struct Template Reference	2773
5.855.1 Detailed Description	2774
5.855.2 Member Typedef Documentation	2774
5.856std::logical_not< void > Struct Template Reference	2774
5.856.1 Detailed Description	2775
5.857std::logical_or< _Tp > Struct Template Reference	2775
5.857.1 Detailed Description	2776
5.857.2 Member Typedef Documentation	2776
5.858std::logical_or< void > Struct Template Reference	2776
5.858.1 Detailed Description	2777
5.859std::lognormal_distribution< _RealType > Class Template Reference	2777
5.859.1 Detailed Description	2778
5.859.2 Member Typedef Documentation	2778
5.859.3 Member Function Documentation	2778
5.859.4 Friends And Related Function Documentation	2779
5.860std::lognormal_distribution< _RealType >::param_type Struct Reference	2781
5.860.1 Detailed Description	2782

5.861std::map< _Key, _Tp, _Compare, _Alloc > Class Template Reference	2782
5.861.1 Detailed Description	2785
5.861.2 Constructor & Destructor Documentation	2785
5.861.3 Member Function Documentation	2788
5.862std::mask_array< _Tp > Class Template Reference	2808
5.862.1 Detailed Description	2809
5.863std::match_results< _Bi_iter, _Alloc > Class Template Reference	2809
5.863.1 Detailed Description	2813
5.863.2 Constructor & Destructor Documentation	2814
5.863.3 Member Function Documentation	2814
5.864std::mem_fun1_ref_t< _Ret, _Tp, _Arg > Class Template Reference	2820
5.864.1 Detailed Description	2820
5.864.2 Member Typedef Documentation	2820
5.865std::mem_fun1_t< _Ret, _Tp, _Arg > Class Template Reference	2821
5.865.1 Detailed Description	2822
5.865.2 Member Typedef Documentation	2822
5.866std::mem_fun_ref_t< _Ret, _Tp > Class Template Reference	2822
5.866.1 Detailed Description	2823
5.866.2 Member Typedef Documentation	2823
5.867std::mem_fun_t< _Ret, _Tp > Class Template Reference	2824
5.867.1 Detailed Description	2824
5.867.2 Member Typedef Documentation	2824
5.868std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f > Class Template Reference	2825
5.868.1 Detailed Description	2826
5.868.2 Member Typedef Documentation	2827
5.868.3 Constructor & Destructor Documentation	2827
5.868.4 Member Function Documentation	2827

5.868.5 Friends And Related Function Documentation	2828
5.869std::messages< _CharT > Class Template Reference	2830
5.869.1 Detailed Description	2831
5.869.2 Member Typedef Documentation	2832
5.869.3 Constructor & Destructor Documentation	2832
5.869.4 Member Function Documentation	2833
5.869.5 Member Data Documentation	2833
5.870std::messages_base Struct Reference	2833
5.870.1 Detailed Description	2834
5.871std::messages_byname< _CharT > Class Template Reference	2834
5.871.1 Detailed Description	2836
5.871.2 Member Function Documentation	2836
5.871.3 Member Data Documentation	2836
5.872std::minus< _Tp > Struct Template Reference	2836
5.872.1 Detailed Description	2837
5.872.2 Member Typedef Documentation	2837
5.873std::minus< void > Struct Template Reference	2838
5.873.1 Detailed Description	2838
5.874std::modulus< _Tp > Struct Template Reference	2838
5.874.1 Detailed Description	2839
5.874.2 Member Typedef Documentation	2839
5.875std::modulus< void > Struct Template Reference	2840
5.875.1 Detailed Description	2840
5.876std::money_base Class Reference	2840
5.876.1 Detailed Description	2841
5.877std::money_get< _CharT, _InIter > Class Template Reference	2841
5.877.1 Detailed Description	2843
5.877.2 Member Typedef Documentation	2843

5.877.3 Constructor & Destructor Documentation	2843
5.877.4 Member Function Documentation	2844
5.877.5 Member Data Documentation	2846
5.878std::money_put< _CharT, _OutIter > Class Template Reference	2846
5.878.1 Detailed Description	2847
5.878.2 Member Typedef Documentation	2848
5.878.3 Constructor & Destructor Documentation	2848
5.878.4 Member Function Documentation	2849
5.878.5 Member Data Documentation	2851
5.879std::money_punct< _CharT, _Intl > Class Template Reference	2851
5.879.1 Detailed Description	2853
5.879.2 Member Typedef Documentation	2853
5.879.3 Constructor & Destructor Documentation	2854
5.879.4 Member Function Documentation	2855
5.879.5 Member Data Documentation	2861
5.880std::money_punct_byname< _CharT, _Intl > Class Template Reference	2861
5.880.1 Detailed Description	2863
5.880.2 Member Function Documentation	2863
5.880.3 Member Data Documentation	2870
5.881std::move_iterator< _Iterator > Class Template Reference	2870
5.881.1 Detailed Description	2871
5.882std::multimap< _Key, _Tp, _Compare, _Alloc > Class Template Reference	2871
5.882.1 Detailed Description	2874
5.882.2 Constructor & Destructor Documentation	2874
5.882.3 Member Function Documentation	2877
5.883std::multiplies< _Tp > Struct Template Reference	2894
5.883.1 Detailed Description	2895
5.883.2 Member Typedef Documentation	2895

5.884std::multiplies< void > Struct Template Reference	2895
5.884.1 Detailed Description	2896
5.885std::multiset< _Key, _Compare, _Alloc > Class Template Reference	2896
5.885.1 Detailed Description	2898
5.885.2 Constructor & Destructor Documentation	2899
5.885.3 Member Function Documentation	2902
5.886std::mutex Class Reference	2916
5.886.1 Detailed Description	2917
5.887std::negate< _Tp > Struct Template Reference	2917
5.887.1 Detailed Description	2918
5.887.2 Member Typedef Documentation	2918
5.888std::negate< void > Struct Template Reference	2918
5.888.1 Detailed Description	2919
5.889std::negative_binomial_distribution< _IntType > Class Template Reference	2919
5.889.1 Detailed Description	2920
5.889.2 Member Typedef Documentation	2920
5.889.3 Member Function Documentation	2920
5.889.4 Friends And Related Function Documentation	2922
5.890std::negative_binomial_distribution< _IntType >::param_type Struct Reference	2923
5.890.1 Detailed Description	2923
5.891std::nested_exception Class Reference	2923
5.891.1 Detailed Description	2924
5.892std::normal_distribution< _RealType > Class Template Reference	2924
5.892.1 Detailed Description	2925
5.892.2 Member Typedef Documentation	2925
5.892.3 Constructor & Destructor Documentation	2925
5.892.4 Member Function Documentation	2926
5.892.5 Friends And Related Function Documentation	2927

5.893std::normal_distribution< _RealType >::param_type Struct Reference	2928
5.893.1 Detailed Description	2928
5.894std::not_equal_to< _Tp > Struct Template Reference	2929
5.894.1 Detailed Description	2929
5.894.2 Member Typedef Documentation	2929
5.895std::not_equal_to< void > Struct Template Reference	2930
5.895.1 Detailed Description	2930
5.896std::num_get< _CharT, _InIter > Class Template Reference	2931
5.896.1 Detailed Description	2933
5.896.2 Member Typedef Documentation	2933
5.896.3 Constructor & Destructor Documentation	2933
5.896.4 Member Function Documentation	2934
5.896.5 Member Data Documentation	2947
5.897std::num_put< _CharT, _OutIter > Class Template Reference	2948
5.897.1 Detailed Description	2949
5.897.2 Member Typedef Documentation	2950
5.897.3 Constructor & Destructor Documentation	2950
5.897.4 Member Function Documentation	2950
5.897.5 Member Data Documentation	2961
5.898std::numeric_limits< _Tp > Struct Template Reference	2961
5.898.1 Detailed Description	2962
5.898.2 Member Function Documentation	2962
5.898.3 Member Data Documentation	2964
5.899std::numeric_limits< bool > Struct Template Reference	2967
5.899.1 Detailed Description	2968
5.900std::numeric_limits< char > Struct Template Reference	2968
5.900.1 Detailed Description	2969
5.901std::numeric_limits< char16_t > Struct Template Reference	2969

5.901.1 Detailed Description	2970
5.902std::numeric_limits< char32_t > Struct Template Reference	2970
5.902.1 Detailed Description	2971
5.903std::numeric_limits< double > Struct Template Reference	2971
5.903.1 Detailed Description	2972
5.904std::numeric_limits< float > Struct Template Reference	2972
5.904.1 Detailed Description	2973
5.905std::numeric_limits< int > Struct Template Reference	2973
5.905.1 Detailed Description	2974
5.906std::numeric_limits< long > Struct Template Reference	2974
5.906.1 Detailed Description	2975
5.907std::numeric_limits< long double > Struct Template Reference	2975
5.907.1 Detailed Description	2976
5.908std::numeric_limits< long long > Struct Template Reference	2976
5.908.1 Detailed Description	2977
5.909std::numeric_limits< short > Struct Template Reference	2977
5.909.1 Detailed Description	2978
5.910std::numeric_limits< signed char > Struct Template Reference	2978
5.910.1 Detailed Description	2979
5.911std::numeric_limits< unsigned char > Struct Template Reference	2979
5.911.1 Detailed Description	2980
5.912std::numeric_limits< unsigned int > Struct Template Reference	2980
5.912.1 Detailed Description	2981
5.913std::numeric_limits< unsigned long > Struct Template Reference	2981
5.913.1 Detailed Description	2982
5.914std::numeric_limits< unsigned long long > Struct Template Reference	2982
5.914.1 Detailed Description	2983
5.915std::numeric_limits< unsigned short > Struct Template Reference	2983

5.915.1 Detailed Description	2984
5.916std::numeric_limits< wchar_t > Struct Template Reference	2984
5.916.1 Detailed Description	2985
5.917std::numpunct< _CharT > Class Template Reference	2986
5.917.1 Detailed Description	2987
5.917.2 Member Typedef Documentation	2988
5.917.3 Constructor & Destructor Documentation	2988
5.917.4 Member Function Documentation	2989
5.917.5 Member Data Documentation	2992
5.918std::numpunct_byname< _CharT > Class Template Reference	2993
5.918.1 Detailed Description	2994
5.918.2 Member Function Documentation	2994
5.918.3 Member Data Documentation	2998
5.919std::once_flag Struct Reference	2998
5.919.1 Detailed Description	2998
5.919.2 Constructor & Destructor Documentation	2998
5.919.3 Member Function Documentation	2999
5.919.4 Friends And Related Function Documentation	2999
5.920std::ostream_iterator< _Tp, _CharT, _Traits > Class Template Reference	2999
5.920.1 Detailed Description	3000
5.920.2 Member Typedef Documentation	3000
5.920.3 Constructor & Destructor Documentation	3001
5.920.4 Member Function Documentation	3002
5.921std::ostreambuf_iterator< _CharT, _Traits > Class Template Reference	3002
5.921.1 Detailed Description	3003
5.921.2 Member Typedef Documentation	3003
5.921.3 Constructor & Destructor Documentation	3005
5.921.4 Member Function Documentation	3005

5.922std::out_of_range Class Reference	3006
5.922.1 Detailed Description	3006
5.922.2 Member Function Documentation	3007
5.923std::output_iterator_tag Struct Reference	3007
5.923.1 Detailed Description	3007
5.924std::overflow_error Class Reference	3007
5.924.1 Detailed Description	3008
5.924.2 Member Function Documentation	3008
5.925std::owner_less< _Tp > Struct Template Reference	3008
5.925.1 Detailed Description	3008
5.926std::owner_less< shared_ptr< _Tp > > Struct Template Reference	3008
5.926.1 Detailed Description	3009
5.926.2 Member Typedef Documentation	3009
5.927std::owner_less< weak_ptr< _Tp > > Struct Template Reference	3010
5.927.1 Detailed Description	3010
5.927.2 Member Typedef Documentation	3010
5.928std::packaged_task< _Res(_ArgTypes...) > Class Template Reference	3011
5.928.1 Detailed Description	3011
5.929std::pair< _T1, _T2 > Struct Template Reference	3011
5.929.1 Detailed Description	3013
5.929.2 Member Typedef Documentation	3013
5.929.3 Constructor & Destructor Documentation	3014
5.929.4 Member Data Documentation	3014
5.930std::piecewise_constant_distribution< _RealType > Class Template Reference	3014
5.930.1 Detailed Description	3015
5.930.2 Member Typedef Documentation	3016
5.930.3 Member Function Documentation	3016
5.930.4 Friends And Related Function Documentation	3017

5.931	std::piecewise_constant_distribution<_RealType>::param_type Struct Reference	3018
5.931.1	Detailed Description	3019
5.932	std::piecewise_construct_t Struct Reference	3019
5.932.1	Detailed Description	3019
5.933	std::piecewise_linear_distribution<_RealType> Class Template Reference	3019
5.933.1	Detailed Description	3020
5.933.2	Member Typedef Documentation	3021
5.933.3	Member Function Documentation	3021
5.933.4	Friends And Related Function Documentation	3022
5.934	std::piecewise_linear_distribution<_RealType>::param_type Struct Reference	3023
5.934.1	Detailed Description	3024
5.935	std::plus<_Tp> Struct Template Reference	3024
5.935.1	Detailed Description	3025
5.935.2	Member Typedef Documentation	3025
5.936	std::pointer_to_binary_function<_Arg1, _Arg2, _Result> Class Template Reference	3026
5.936.1	Detailed Description	3026
5.936.2	Member Typedef Documentation	3027
5.937	std::pointer_to_unary_function<_Arg, _Result> Class Template Reference	3027
5.937.1	Detailed Description	3028
5.937.2	Member Typedef Documentation	3028
5.938	std::pointer_traits<_Ptr> Struct Template Reference	3029
5.938.1	Detailed Description	3029
5.938.2	Member Typedef Documentation	3029
5.939	std::pointer_traits<_Tp*> Struct Template Reference	3030
5.939.1	Detailed Description	3030
5.939.2	Member Typedef Documentation	3030
5.939.3	Member Function Documentation	3031
5.940	std::poisson_distribution<_IntType> Class Template Reference	3031

5.940.1 Detailed Description	3032
5.940.2 Member Typedef Documentation	3033
5.940.3 Member Function Documentation	3033
5.940.4 Friends And Related Function Documentation	3034
5.941std::poisson_distribution< _IntType >::param_type Struct Reference	3035
5.941.1 Detailed Description	3036
5.942std::priority_queue< _Tp, _Sequence, _Compare > Class Template Reference	3036
5.942.1 Detailed Description	3037
5.942.2 Constructor & Destructor Documentation	3038
5.942.3 Member Function Documentation	3039
5.943std::promise< _Res > Class Template Reference	3040
5.943.1 Detailed Description	3041
5.944std::promise< _Res & > Class Template Reference	3041
5.944.1 Detailed Description	3042
5.945std::promise< void > Class Template Reference	3042
5.945.1 Detailed Description	3042
5.946std::queue< _Tp, _Sequence > Class Template Reference	3043
5.946.1 Detailed Description	3043
5.946.2 Constructor & Destructor Documentation	3044
5.946.3 Member Function Documentation	3044
5.946.4 Member Data Documentation	3046
5.947std::random_access_iterator_tag Struct Reference	3046
5.947.1 Detailed Description	3047
5.948std::random_device Class Reference	3047
5.948.1 Detailed Description	3047
5.948.2 Member Typedef Documentation	3047
5.949std::range_error Class Reference	3048
5.949.1 Detailed Description	3048

5.949.2 Member Function Documentation	3048
5.950std::ratio< _Num, _Den > Struct Template Reference	3049
5.950.1 Detailed Description	3049
5.951std::ratio_equal< _R1, _R2 > Struct Template Reference	3049
5.951.1 Detailed Description	3050
5.952std::ratio_not_equal< _R1, _R2 > Struct Template Reference	3050
5.952.1 Detailed Description	3051
5.953std::raw_storage_iterator< _OutputIterator, _Tp > Class Template Reference	3051
5.953.1 Detailed Description	3052
5.953.2 Member Typedef Documentation	3052
5.954std::recursive_mutex Class Reference	3053
5.954.1 Detailed Description	3054
5.955std::recursive_timed_mutex Class Reference	3054
5.955.1 Detailed Description	3054
5.956std::reference_wrapper< _Tp > Class Template Reference	3054
5.956.1 Detailed Description	3055
5.957std::regex_error Class Reference	3055
5.957.1 Detailed Description	3056
5.957.2 Constructor & Destructor Documentation	3056
5.957.3 Member Function Documentation	3056
5.958std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits > Class Template Reference	3057
5.958.1 Detailed Description	3057
5.958.2 Constructor & Destructor Documentation	3057
5.958.3 Member Function Documentation	3058
5.959std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits > Class Template Reference	3059
5.959.1 Detailed Description	3060
5.959.2 Constructor & Destructor Documentation	3060
5.959.3 Member Function Documentation	3062

5.960std::regex_traits<_Ch_type> Struct Template Reference	3064
5.960.1 Detailed Description	3065
5.960.2 Constructor & Destructor Documentation	3065
5.960.3 Member Function Documentation	3065
5.961std::reverse_iterator<_Iterator> Class Template Reference	3071
5.961.1 Detailed Description	3072
5.961.2 Member Typedef Documentation	3072
5.961.3 Constructor & Destructor Documentation	3073
5.961.4 Member Function Documentation	3073
5.962std::runtime_error Class Reference	3077
5.962.1 Detailed Description	3077
5.962.2 Constructor & Destructor Documentation	3077
5.962.3 Member Function Documentation	3078
5.963std::scoped_allocator_adaptor<_OuterAlloc, _InnerAllocs> Class Template Reference	3078
5.963.1 Detailed Description	3080
5.964std::seed_seq Class Reference	3080
5.964.1 Detailed Description	3080
5.964.2 Member Typedef Documentation	3080
5.964.3 Constructor & Destructor Documentation	3081
5.965std::set<_Key, _Compare, _Alloc> Class Template Reference	3081
5.965.1 Detailed Description	3083
5.965.2 Member Typedef Documentation	3084
5.965.3 Constructor & Destructor Documentation	3086
5.965.4 Member Function Documentation	3089
5.966std::shared_future<_Res> Class Template Reference	3106
5.966.1 Detailed Description	3108
5.966.2 Member Typedef Documentation	3108
5.966.3 Constructor & Destructor Documentation	3108

5.966.4 Member Function Documentation	3108
5.967std::shared_future< _Res & > Class Template Reference	3109
5.967.1 Detailed Description	3110
5.967.2 Member Typedef Documentation	3110
5.967.3 Constructor & Destructor Documentation	3110
5.967.4 Member Function Documentation	3111
5.968std::shared_future< void > Class Template Reference	3111
5.968.1 Detailed Description	3112
5.968.2 Member Typedef Documentation	3113
5.968.3 Constructor & Destructor Documentation	3113
5.968.4 Member Function Documentation	3113
5.969std::shared_lock< _Mutex > Class Template Reference	3114
5.969.1 Detailed Description	3114
5.970std::shared_ptr< _Tp > Class Template Reference	3115
5.970.1 Detailed Description	3116
5.970.2 Constructor & Destructor Documentation	3116
5.970.3 Friends And Related Function Documentation	3122
5.971std::shared_timed_mutex Class Reference	3123
5.971.1 Detailed Description	3123
5.972std::shuffle_order_engine< _RandomNumberEngine, __k > Class Template Reference	3124
5.972.1 Detailed Description	3125
5.972.2 Member Typedef Documentation	3125
5.972.3 Constructor & Destructor Documentation	3125
5.972.4 Member Function Documentation	3126
5.972.5 Friends And Related Function Documentation	3128
5.973std::slice Class Reference	3129
5.973.1 Detailed Description	3129
5.974std::slice_array< _Tp > Class Template Reference	3130

5.974.1 Detailed Description	3131
5.975std::stack< _Tp, _Sequence > Class Template Reference	3131
5.975.1 Detailed Description	3132
5.975.2 Constructor & Destructor Documentation	3133
5.975.3 Member Function Documentation	3133
5.976std::student_t_distribution< _RealType > Class Template Reference	3134
5.976.1 Detailed Description	3135
5.976.2 Member Typedef Documentation	3136
5.976.3 Member Function Documentation	3136
5.976.4 Friends And Related Function Documentation	3137
5.977std::student_t_distribution< _RealType >::param_type Struct Reference	3138
5.977.1 Detailed Description	3138
5.978std::sub_match< _Bilter > Class Template Reference	3139
5.978.1 Detailed Description	3140
5.978.2 Member Typedef Documentation	3140
5.978.3 Member Function Documentation	3140
5.978.4 Member Data Documentation	3142
5.979std::subtract_with_carry_engine< _UIntType, __w, __s, __r > Class Template Reference	3143
5.979.1 Detailed Description	3144
5.979.2 Member Typedef Documentation	3144
5.979.3 Constructor & Destructor Documentation	3144
5.979.4 Member Function Documentation	3145
5.979.5 Friends And Related Function Documentation	3146
5.980std::system_error Class Reference	3148
5.980.1 Detailed Description	3148
5.980.2 Member Function Documentation	3148
5.981std::thread Class Reference	3149
5.981.1 Detailed Description	3149

5.981.2 Member Function Documentation	3149
5.982std::thread::id Class Reference	3150
5.982.1 Detailed Description	3150
5.983std::time_base Class Reference	3150
5.983.1 Detailed Description	3151
5.984std::time_get< _CharT, _InIter > Class Template Reference	3151
5.984.1 Detailed Description	3153
5.984.2 Member Typedef Documentation	3153
5.984.3 Constructor & Destructor Documentation	3153
5.984.4 Member Function Documentation	3154
5.984.5 Member Data Documentation	3162
5.985std::time_get_byname< _CharT, _InIter > Class Template Reference	3163
5.985.1 Detailed Description	3164
5.985.2 Member Function Documentation	3165
5.985.3 Member Data Documentation	3173
5.986std::time_put< _CharT, _OutIter > Class Template Reference	3174
5.986.1 Detailed Description	3175
5.986.2 Member Typedef Documentation	3175
5.986.3 Constructor & Destructor Documentation	3175
5.986.4 Member Function Documentation	3176
5.986.5 Member Data Documentation	3178
5.987std::time_put_byname< _CharT, _OutIter > Class Template Reference	3178
5.987.1 Detailed Description	3179
5.987.2 Member Function Documentation	3179
5.987.3 Member Data Documentation	3181
5.988std::timed_mutex Class Reference	3181
5.988.1 Detailed Description	3182
5.989std::tr2::__dynamic_bitset_base< _WordT, _Alloc > Struct Template Reference	3182

5.989.1 Detailed Description	3184
5.989.2 Member Data Documentation	3184
5.990std::tr2::__reflection_typelist< _Elements > Struct Template Reference	3184
5.990.1 Detailed Description	3184
5.991std::tr2::__reflection_typelist< _First, _Rest... > Struct Template Reference	3184
5.991.1 Detailed Description	3185
5.992std::tr2::__reflection_typelist<> Struct Template Reference	3185
5.992.1 Detailed Description	3185
5.993std::tr2::bases< _Tp > Struct Template Reference	3185
5.993.1 Detailed Description	3186
5.994std::tr2::bool_set Class Reference	3186
5.994.1 Detailed Description	3187
5.994.2 Constructor & Destructor Documentation	3187
5.994.3 Member Function Documentation	3187
5.995std::tr2::direct_bases< _Tp > Struct Template Reference	3188
5.995.1 Detailed Description	3188
5.996std::tr2::dynamic_bitset< _WordT, _Alloc > Class Template Reference	3189
5.996.1 Detailed Description	3192
5.996.2 Constructor & Destructor Documentation	3193
5.996.3 Member Function Documentation	3195
5.997std::tr2::dynamic_bitset< _WordT, _Alloc >::reference Class Reference	3205
5.997.1 Detailed Description	3206
5.998std::try_to_lock_t Struct Reference	3206
5.998.1 Detailed Description	3206
5.999std::tuple< _Elements > Class Template Reference	3206
5.999.1 Detailed Description	3208
5.1000std::tuple< _T1, _T2 > Class Template Reference	3209
5.1000.1 Detailed Description	3211

5.100	<code>std::tuple_element< _Int, _Tp ></code> Struct Template Reference	3211
5.1001.	Detailed Description	3211
5.1002	<code>std::tuple_element< 0, std::pair< _Tp1, _Tp2 > ></code> Struct Template Reference	3212
5.1002.	Detailed Description	3212
5.1003	<code>std::tuple_element< 0, tuple< _Head, _Tail... > ></code> Struct Template Reference	3212
5.1003.	Detailed Description	3212
5.1004	<code>std::tuple_element< 1, std::pair< _Tp1, _Tp2 > ></code> Struct Template Reference	3212
5.1004.	Detailed Description	3213
5.1005	<code>std::tuple_element< __i, tuple< _Head, _Tail... > ></code> Struct Template Reference	3213
5.1005.	Detailed Description	3213
5.1006	<code>std::tuple_element< _Int, std::__debug::array< _Tp, _Nm > ></code> Struct Template Reference	3214
5.1006.	Detailed Description	3214
5.1007	<code>std::tuple_element< _Int, ::array< _Tp, _Nm > ></code> Struct Template Reference	3214
5.1007.	Detailed Description	3214
5.1008	<code>std::tuple_size< _Tp ></code> Struct Template Reference	3215
5.1008.	Detailed Description	3215
5.1009	<code>std::tuple_size< std::__debug::array< _Tp, _Nm > ></code> Struct Template Reference	3215
5.1009.	Detailed Description	3216
5.1010	<code>std::tuple_size< std::pair< _Tp1, _Tp2 > ></code> Struct Template Reference	3216
5.1010.	Detailed Description	3217
5.1011	<code>std::tuple_size< tuple< _Elements... > ></code> Struct Template Reference	3217
5.1011.	Detailed Description	3218
5.1012	<code>std::tuple_size< ::array< _Tp, _Nm > ></code> Struct Template Reference	3218
5.1012.	Detailed Description	3219
5.1013	<code>std::type_index</code> Struct Reference	3219
5.1013.	Detailed Description	3219
5.1014	<code>std::type_info</code> Class Reference	3220
5.1014.	Detailed Description	3220

5.1014.2	Constructor & Destructor Documentation	3220
5.1014.3	Member Function Documentation	3221
5.1015	std::unary_function< _Arg, _Result > Struct Template Reference	3221
5.1015.1	Detailed Description	3221
5.1015.2	Member Typedef Documentation	3222
5.1016	std::unary_negate< _Predicate > Class Template Reference	3222
5.1016.1	Detailed Description	3223
5.1016.2	Member Typedef Documentation	3223
5.1017	std::underflow_error Class Reference	3224
5.1017.1	Detailed Description	3224
5.1017.2	Member Function Documentation	3224
5.1018	std::uniform_int_distribution< _IntType > Class Template Reference	3225
5.1018.1	Detailed Description	3225
5.1018.2	Member Typedef Documentation	3226
5.1018.3	Constructor & Destructor Documentation	3226
5.1018.4	Member Function Documentation	3226
5.1018.5	Friends And Related Function Documentation	3227
5.1019	std::uniform_int_distribution< _IntType >::param_type Struct Reference	3227
5.1019.1	Detailed Description	3228
5.1020	std::uniform_real_distribution< _RealType > Class Template Reference	3228
5.1020.1	Detailed Description	3229
5.1020.2	Member Typedef Documentation	3229
5.1020.3	Constructor & Destructor Documentation	3229
5.1020.4	Member Function Documentation	3229
5.1020.5	Friends And Related Function Documentation	3231
5.1021	std::uniform_real_distribution< _RealType >::param_type Struct Reference	3231
5.1021.1	Detailed Description	3231
5.1022	std::unique_lock< _Mutex > Class Template Reference	3232

5.1022. Detailed Description	3232
5.1023. std::unique_ptr< _Tp, _Dp > Class Template Reference	3233
5.1023.1. Detailed Description	3233
5.1023.2. Constructor & Destructor Documentation	3234
5.1023.3. Member Function Documentation	3236
5.1024. std::unique_ptr< _Tp[], _Dp > Class Template Reference	3238
5.1024.1. Detailed Description	3239
5.1024.2. Constructor & Destructor Documentation	3239
5.1024.3. Member Function Documentation	3241
5.1025. std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > Class Template Reference	3243
5.1025.1. Detailed Description	3246
5.1025.2. Member Typedef Documentation	3246
5.1025.3. Constructor & Destructor Documentation	3249
5.1025.4. Member Function Documentation	3251
5.1026. std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > Class Template Reference	3267
5.1026.1. Detailed Description	3269
5.1026.2. Member Typedef Documentation	3270
5.1026.3. Constructor & Destructor Documentation	3273
5.1026.4. Member Function Documentation	3275
5.1027. std::unordered_multiset< _Value, _Hash, _Pred, _Alloc > Class Template Reference	3289
5.1027.1. Detailed Description	3291
5.1027.2. Member Typedef Documentation	3292
5.1027.3. Constructor & Destructor Documentation	3294
5.1027.4. Member Function Documentation	3296
5.1028. std::unordered_set< _Value, _Hash, _Pred, _Alloc > Class Template Reference	3311
5.1028.1. Detailed Description	3313
5.1028.2. Member Typedef Documentation	3314
5.1028.3. Constructor & Destructor Documentation	3316

5.1028.4	Member Function Documentation	3318
5.1029	std::uses_allocator< _Tp, _Alloc > Struct Template Reference	3334
5.1029.1	Detailed Description	3334
5.1030	std::uses_allocator< tuple< _Types... >, _Alloc > Struct Template Reference	3334
5.1030.1	Detailed Description	3335
5.1031	std::valarray< _Tp > Class Template Reference	3335
5.1031.1	Detailed Description	3337
5.1031.2	Constructor & Destructor Documentation	3337
5.1032	std::vector< _Tp, _Alloc > Class Template Reference	3338
5.1032.1	Detailed Description	3341
5.1032.2	Constructor & Destructor Documentation	3341
5.1032.3	Member Function Documentation	3344
5.1033	std::vector< bool, _Alloc > Class Template Reference	3358
5.1033.1	Detailed Description	3361
5.1034	std::wbuffer_convert< _Codecvt, _Elem, _Tr > Class Template Reference	3362
5.1034.1	Detailed Description	3364
5.1034.2	Member Typedef Documentation	3364
5.1034.3	Constructor & Destructor Documentation	3364
5.1034.4	Member Function Documentation	3365
5.1034.5	Member Data Documentation	3377
5.1035	std::weak_ptr< _Tp > Class Template Reference	3378
5.1035.1	Detailed Description	3379
5.1036	std::weibull_distribution< _RealType > Class Template Reference	3379
5.1036.1	Detailed Description	3380
5.1036.2	Member Typedef Documentation	3381
5.1036.3	Member Function Documentation	3381
5.1036.4	Friends And Related Function Documentation	3383
5.1037	std::weibull_distribution< _RealType >::param_type Struct Reference	3383
5.1037.1	Detailed Description	3384
5.1038	std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc > Class Template Reference	3384
5.1038.1	Detailed Description	3385
5.1038.2	Constructor & Destructor Documentation	3385
5.1038.3	Member Function Documentation	3386

6 File Documentation	3388
6.1 algo.h File Reference	3388
6.1.1 Detailed Description	3398
6.2 algobase.h File Reference	3398
6.2.1 Detailed Description	3400
6.3 algorithm File Reference	3400
6.3.1 Detailed Description	3400
6.4 algorithm File Reference	3400
6.4.1 Detailed Description	3401
6.5 algorithm File Reference	3402
6.5.1 Detailed Description	3402
6.6 algorithm File Reference	3402
6.6.1 Detailed Description	3402
6.6.2 Function Documentation	3403
6.7 algorithmfwd.h File Reference	3403
6.7.1 Detailed Description	3408
6.8 algorithmfwd.h File Reference	3409
6.8.1 Detailed Description	3417
6.9 aligned_buffer.h File Reference	3417
6.9.1 Detailed Description	3417
6.10 alloc_traits.h File Reference	3417
6.10.1 Detailed Description	3418
6.11 alloc_traits.h File Reference	3418
6.11.1 Detailed Description	3418
6.12 allocated_ptr.h File Reference	3419
6.12.1 Detailed Description	3419
6.13 allocator.h File Reference	3419
6.13.1 Detailed Description	3420

6.14	any File Reference	3420
6.14.1	Detailed Description	3420
6.15	array File Reference	3421
6.15.1	Detailed Description	3421
6.16	array File Reference	3422
6.16.1	Detailed Description	3422
6.17	array File Reference	3423
6.17.1	Detailed Description	3423
6.18	array_allocator.h File Reference	3423
6.18.1	Detailed Description	3424
6.19	assertions.h File Reference	3424
6.19.1	Detailed Description	3424
6.20	assoc_container.hpp File Reference	3424
6.20.1	Detailed Description	3425
6.21	atomic File Reference	3425
6.21.1	Detailed Description	3429
6.22	atomic_base.h File Reference	3429
6.22.1	Detailed Description	3430
6.23	atomic_futex.h File Reference	3430
6.23.1	Detailed Description	3430
6.24	atomic_lockfree_defines.h File Reference	3430
6.24.1	Detailed Description	3431
6.25	atomic_word.h File Reference	3431
6.25.1	Detailed Description	3431
6.26	atomicity.h File Reference	3431
6.26.1	Detailed Description	3432
6.27	auto_ptr.h File Reference	3432
6.27.1	Detailed Description	3432

6.28	backward_warning.h File Reference	3432
6.28.1	Detailed Description	3432
6.29	balanced_quicksort.h File Reference	3432
6.29.1	Detailed Description	3433
6.30	base.h File Reference	3433
6.30.1	Detailed Description	3433
6.31	base.h File Reference	3433
6.31.1	Detailed Description	3434
6.32	basic_file.h File Reference	3434
6.32.1	Detailed Description	3434
6.33	basic_ios.h File Reference	3435
6.33.1	Detailed Description	3435
6.34	basic_ios.tcc File Reference	3435
6.34.1	Detailed Description	3435
6.35	basic_iterator.h File Reference	3435
6.35.1	Detailed Description	3435
6.36	basic_string.h File Reference	3436
6.36.1	Detailed Description	3438
6.37	basic_string.tcc File Reference	3438
6.37.1	Detailed Description	3439
6.38	bin_search_tree_.hpp File Reference	3439
6.38.1	Detailed Description	3439
6.39	binary_heap_.hpp File Reference	3440
6.39.1	Detailed Description	3440
6.40	binders.h File Reference	3440
6.40.1	Detailed Description	3441
6.41	binomial_heap_.hpp File Reference	3441
6.41.1	Detailed Description	3441

6.42	binomial_heap_base.hpp File Reference	3441
6.42.1	Detailed Description	3442
6.43	bitmap_allocator.h File Reference	3442
6.43.1	Detailed Description	3443
6.43.2	Macro Definition Documentation	3443
6.44	bitset File Reference	3443
6.44.1	Detailed Description	3444
6.45	bitset File Reference	3444
6.45.1	Detailed Description	3444
6.46	bitset File Reference	3444
6.46.1	Detailed Description	3445
6.47	bool_set File Reference	3445
6.47.1	Detailed Description	3446
6.48	bool_set.tcc File Reference	3446
6.48.1	Detailed Description	3446
6.49	boost_concept_check.h File Reference	3447
6.49.1	Detailed Description	3447
6.50	branch_policy.hpp File Reference	3447
6.50.1	Detailed Description	3447
6.51	c++0x_warning.h File Reference	3448
6.51.1	Detailed Description	3448
6.52	c++14_warning.h File Reference	3448
6.52.1	Detailed Description	3448
6.53	c++allocator.h File Reference	3448
6.53.1	Detailed Description	3448
6.54	c++config.h File Reference	3448
6.54.1	Detailed Description	3454
6.55	c++io.h File Reference	3454

6.55.1 Detailed Description	3454
6.56 c++locale.h File Reference	3454
6.56.1 Detailed Description	3455
6.57 c++locale_internal.h File Reference	3455
6.57.1 Detailed Description	3455
6.58 cassert File Reference	3455
6.58.1 Detailed Description	3455
6.59 cast.h File Reference	3456
6.59.1 Detailed Description	3456
6.60 cc_hash_max_collision_check_resize_trigger_imp.hpp File Reference	3456
6.60.1 Detailed Description	3456
6.61 cc_ht_map_.hpp File Reference	3456
6.61.1 Detailed Description	3457
6.62 ccomplex File Reference	3457
6.62.1 Detailed Description	3457
6.63 ccomplex File Reference	3457
6.63.1 Detailed Description	3457
6.64 ctype File Reference	3457
6.64.1 Detailed Description	3458
6.65 ctype File Reference	3458
6.65.1 Detailed Description	3458
6.66 cerrno File Reference	3458
6.66.1 Detailed Description	3458
6.67 cenv File Reference	3458
6.67.1 Detailed Description	3459
6.68 cenv File Reference	3459
6.68.1 Detailed Description	3459
6.69 cfloat File Reference	3459

6.69.1 Detailed Description	3459
6.70 cfloat File Reference	3459
6.70.1 Detailed Description	3459
6.71 char_traits.h File Reference	3460
6.71.1 Detailed Description	3460
6.72 checkers.h File Reference	3460
6.72.1 Detailed Description	3460
6.73 chrono File Reference	3460
6.73.1 Detailed Description	3463
6.73.2 Typedef Documentation	3463
6.74 chrono File Reference	3463
6.74.1 Detailed Description	3464
6.75 cinttypes File Reference	3464
6.75.1 Detailed Description	3464
6.76 cinttypes File Reference	3464
6.76.1 Detailed Description	3464
6.77 ciso646 File Reference	3464
6.77.1 Detailed Description	3464
6.78 climits File Reference	3465
6.78.1 Detailed Description	3465
6.79 climits File Reference	3465
6.79.1 Detailed Description	3465
6.80 locale File Reference	3465
6.80.1 Detailed Description	3465
6.81 cmath File Reference	3466
6.81.1 Detailed Description	3468
6.82 cmath File Reference	3468
6.82.1 Detailed Description	3468

6.83	cmath File Reference	3468
6.83.1	Detailed Description	3471
6.84	cmp_fn_imps.hpp File Reference	3471
6.84.1	Detailed Description	3471
6.85	codecvt File Reference	3471
6.85.1	Detailed Description	3471
6.86	codecvt.h File Reference	3471
6.86.1	Detailed Description	3472
6.87	codecvt_specializations.h File Reference	3472
6.87.1	Detailed Description	3472
6.88	compatibility.h File Reference	3472
6.88.1	Detailed Description	3472
6.89	compatibility.h File Reference	3472
6.89.1	Detailed Description	3473
6.90	compiletime_settings.h File Reference	3473
6.90.1	Detailed Description	3473
6.90.2	Macro Definition Documentation	3473
6.91	complex File Reference	3474
6.91.1	Detailed Description	3478
6.92	complex File Reference	3478
6.92.1	Detailed Description	3479
6.93	complex.h File Reference	3479
6.93.1	Detailed Description	3479
6.94	concept_check.h File Reference	3480
6.94.1	Detailed Description	3480
6.95	concurrency.h File Reference	3480
6.95.1	Detailed Description	3480
6.96	cond_dealtor.hpp File Reference	3481

6.96.1 Detailed Description	3481
6.97 cond_key_dtor_entry_dealtor.hpp File Reference	3481
6.97.1 Detailed Description	3481
6.98 condition_variable File Reference	3481
6.98.1 Detailed Description	3482
6.99 const_iterator.hpp File Reference	3482
6.99.1 Detailed Description	3482
6.100const_iterator.hpp File Reference	3482
6.100.1 Detailed Description	3483
6.101const_iterator.hpp File Reference	3483
6.101.1 Detailed Description	3483
6.102constructor_destructor_fn_imps.hpp File Reference	3483
6.102.1 Detailed Description	3483
6.103constructor_destructor_fn_imps.hpp File Reference	3483
6.103.1 Detailed Description	3483
6.104constructor_destructor_fn_imps.hpp File Reference	3483
6.105constructor_destructor_no_store_hash_fn_imps.hpp File Reference	3483
6.105.1 Detailed Description	3483
6.106constructor_destructor_no_store_hash_fn_imps.hpp File Reference	3484
6.106.1 Detailed Description	3484
6.107constructor_destructor_store_hash_fn_imps.hpp File Reference	3484
6.107.1 Detailed Description	3484
6.108constructor_destructor_store_hash_fn_imps.hpp File Reference	3484
6.108.1 Detailed Description	3484
6.109constructors_destructor_fn_imps.hpp File Reference	3484
6.109.1 Detailed Description	3484
6.110constructors_destructor_fn_imps.hpp File Reference	3484
6.110.1 Detailed Description	3484

6.111	constructors_destructor_fn_impls.hpp File Reference	3484
6.111.1	Detailed Description	3484
6.112	constructors_destructor_fn_impls.hpp File Reference	3485
6.112.1	Detailed Description	3485
6.113	constructors_destructor_fn_impls.hpp File Reference	3485
6.113.1	Detailed Description	3485
6.114	constructors_destructor_fn_impls.hpp File Reference	3485
6.114.1	Detailed Description	3485
6.115	constructors_destructor_fn_impls.hpp File Reference	3485
6.115.1	Detailed Description	3485
6.116	constructors_destructor_fn_impls.hpp File Reference	3485
6.116.1	Detailed Description	3485
6.117	constructors_destructor_fn_impls.hpp File Reference	3485
6.117.1	Detailed Description	3485
6.118	constructors_destructor_fn_impls.hpp File Reference	3486
6.118.1	Detailed Description	3486
6.119	constructors_destructor_fn_impls.hpp File Reference	3486
6.119.1	Detailed Description	3486
6.120	constructors_destructor_fn_impls.hpp File Reference	3486
6.120.1	Detailed Description	3486
6.121	container_base_dispatch.hpp File Reference	3486
6.121.1	Detailed Description	3487
6.122	cpp_type_traits.h File Reference	3487
6.122.1	Detailed Description	3487
6.123	cpu_defines.h File Reference	3487
6.123.1	Detailed Description	3487
6.124	csetjmp File Reference	3488
6.124.1	Detailed Description	3488

6.125csignal File Reference	3488
6.125.1 Detailed Description	3488
6.126cstdalign File Reference	3488
6.126.1 Detailed Description	3489
6.127cstdarg File Reference	3489
6.127.1 Detailed Description	3489
6.128cstdarg File Reference	3489
6.128.1 Detailed Description	3489
6.129cstdbool File Reference	3489
6.129.1 Detailed Description	3489
6.130cstdbool File Reference	3490
6.130.1 Detailed Description	3490
6.131cstddef File Reference	3490
6.131.1 Detailed Description	3490
6.132cstdint File Reference	3490
6.132.1 Detailed Description	3490
6.133cstdint File Reference	3491
6.133.1 Detailed Description	3491
6.134cstdio File Reference	3491
6.134.1 Detailed Description	3491
6.135cstdio File Reference	3491
6.135.1 Detailed Description	3491
6.136cstdlib File Reference	3492
6.136.1 Detailed Description	3492
6.137cstdlib File Reference	3492
6.137.1 Detailed Description	3492
6.138cstring File Reference	3492
6.138.1 Detailed Description	3493

6.139ctgmth File Reference	3493
6.139.1 Detailed Description	3493
6.140ctgmth File Reference	3493
6.140.1 Detailed Description	3493
6.141ctime File Reference	3494
6.141.1 Detailed Description	3494
6.142ctime File Reference	3494
6.142.1 Detailed Description	3494
6.143ctype_base.h File Reference	3494
6.143.1 Detailed Description	3494
6.144ctype_inline.h File Reference	3495
6.144.1 Detailed Description	3495
6.145cuchar File Reference	3495
6.145.1 Detailed Description	3495
6.146cwchar File Reference	3495
6.146.1 Detailed Description	3496
6.147cwchar File Reference	3496
6.147.1 Detailed Description	3496
6.148cwctype File Reference	3496
6.148.1 Detailed Description	3496
6.149cwctype File Reference	3497
6.149.1 Detailed Description	3497
6.150cxxabi.h File Reference	3497
6.150.1 Detailed Description	3498
6.150.2 Function Documentation	3498
6.151cxxabi_forced.h File Reference	3499
6.151.1 Detailed Description	3499
6.152cxxabi_tweaks.h File Reference	3499

6.152.1 Detailed Description	3500
6.153debug.h File Reference	3500
6.153.1 Detailed Description	3501
6.154debug_allocator.h File Reference	3501
6.154.1 Detailed Description	3501
6.155debug_fn_imps.hpp File Reference	3501
6.155.1 Detailed Description	3501
6.156debug_fn_imps.hpp File Reference	3501
6.156.1 Detailed Description	3501
6.157debug_fn_imps.hpp File Reference	3501
6.157.1 Detailed Description	3501
6.158debug_fn_imps.hpp File Reference	3502
6.158.1 Detailed Description	3502
6.159debug_fn_imps.hpp File Reference	3502
6.159.1 Detailed Description	3502
6.160debug_fn_imps.hpp File Reference	3502
6.160.1 Detailed Description	3502
6.161debug_fn_imps.hpp File Reference	3502
6.161.1 Detailed Description	3502
6.162debug_fn_imps.hpp File Reference	3502
6.162.1 Detailed Description	3502
6.163debug_fn_imps.hpp File Reference	3502
6.163.1 Detailed Description	3502
6.164debug_fn_imps.hpp File Reference	3503
6.164.1 Detailed Description	3503
6.165debug_fn_imps.hpp File Reference	3503
6.165.1 Detailed Description	3503
6.166debug_fn_imps.hpp File Reference	3503

6.166.1 Detailed Description	3503
6.167debug_fn_imps.hpp File Reference	3503
6.167.1 Detailed Description	3503
6.168debug_fn_imps.hpp File Reference	3503
6.168.1 Detailed Description	3503
6.169debug_fn_imps.hpp File Reference	3503
6.169.1 Detailed Description	3503
6.170debug_map_base.hpp File Reference	3504
6.170.1 Detailed Description	3504
6.171debug_no_store_hash_fn_imps.hpp File Reference	3504
6.171.1 Detailed Description	3504
6.172debug_no_store_hash_fn_imps.hpp File Reference	3504
6.172.1 Detailed Description	3504
6.173debug_store_hash_fn_imps.hpp File Reference	3504
6.173.1 Detailed Description	3504
6.174debug_store_hash_fn_imps.hpp File Reference	3504
6.174.1 Detailed Description	3504
6.175decimal File Reference	3504
6.175.1 Detailed Description	3514
6.176deque File Reference	3514
6.176.1 Detailed Description	3514
6.177deque File Reference	3514
6.177.1 Detailed Description	3515
6.178deque File Reference	3515
6.178.1 Detailed Description	3515
6.179deque File Reference	3515
6.179.1 Detailed Description	3516
6.180deque.tcc File Reference	3516

6.180.1 Detailed Description	3517
6.181direct_mask_range_hashing_imp.hpp File Reference	3517
6.181.1 Detailed Description	3517
6.182direct_mod_range_hashing_imp.hpp File Reference	3517
6.182.1 Detailed Description	3517
6.183dynamic_bitset File Reference	3517
6.183.1 Detailed Description	3518
6.184dynamic_bitset.tcc File Reference	3518
6.184.1 Detailed Description	3519
6.185enable_special_members.h File Reference	3519
6.185.1 Detailed Description	3519
6.186enc_filebuf.h File Reference	3519
6.186.1 Detailed Description	3519
6.187entry_cmp.hpp File Reference	3520
6.187.1 Detailed Description	3520
6.188entry_list_fn_imps.hpp File Reference	3520
6.188.1 Detailed Description	3520
6.189entry_metadata_base.hpp File Reference	3520
6.189.1 Detailed Description	3520
6.190entry_pred.hpp File Reference	3520
6.190.1 Detailed Description	3521
6.191eq_by_less.hpp File Reference	3521
6.191.1 Detailed Description	3521
6.192equally_split.h File Reference	3521
6.192.1 Detailed Description	3521
6.193erase_fn_imps.hpp File Reference	3522
6.193.1 Detailed Description	3522
6.194erase_fn_imps.hpp File Reference	3522

6.194.1 Detailed Description	3522
6.195erase_fn_imps.hpp File Reference	3522
6.195.1 Detailed Description	3522
6.196erase_fn_imps.hpp File Reference	3522
6.196.1 Detailed Description	3522
6.197erase_fn_imps.hpp File Reference	3522
6.197.1 Detailed Description	3522
6.198erase_fn_imps.hpp File Reference	3522
6.198.1 Detailed Description	3522
6.199erase_fn_imps.hpp File Reference	3523
6.199.1 Detailed Description	3523
6.200erase_fn_imps.hpp File Reference	3523
6.200.1 Detailed Description	3523
6.201erase_fn_imps.hpp File Reference	3523
6.201.1 Detailed Description	3523
6.202erase_fn_imps.hpp File Reference	3523
6.202.1 Detailed Description	3523
6.203erase_fn_imps.hpp File Reference	3523
6.203.1 Detailed Description	3523
6.204erase_fn_imps.hpp File Reference	3523
6.204.1 Detailed Description	3523
6.205erase_fn_imps.hpp File Reference	3524
6.205.1 Detailed Description	3524
6.206erase_fn_imps.hpp File Reference	3524
6.206.1 Detailed Description	3524
6.207erase_if.h File Reference	3524
6.207.1 Detailed Description	3524
6.208erase_no_store_hash_fn_imps.hpp File Reference	3524

6.208.1 Detailed Description	3524
6.209erase_no_store_hash_fn_imps.hpp File Reference	3524
6.209.1 Detailed Description	3524
6.210erase_store_hash_fn_imps.hpp File Reference	3525
6.210.1 Detailed Description	3525
6.211erase_store_hash_fn_imps.hpp File Reference	3525
6.211.1 Detailed Description	3525
6.212error_constants.h File Reference	3525
6.212.1 Detailed Description	3525
6.213exception File Reference	3526
6.213.1 Detailed Description	3526
6.214exception.hpp File Reference	3526
6.214.1 Detailed Description	3527
6.215exception_defines.h File Reference	3527
6.215.1 Detailed Description	3527
6.216exception_ptr.h File Reference	3527
6.216.1 Detailed Description	3528
6.217extc++.h File Reference	3528
6.217.1 Detailed Description	3528
6.218extptr_allocator.h File Reference	3528
6.218.1 Detailed Description	3529
6.219features.h File Reference	3529
6.219.1 Detailed Description	3529
6.219.2 Macro Definition Documentation	3529
6.220fenv.h File Reference	3531
6.220.1 Detailed Description	3531
6.221filesystem File Reference	3531
6.221.1 Detailed Description	3532

6.222find.h File Reference	3532
6.222.1 Detailed Description	3532
6.223find_fn_imps.hpp File Reference	3532
6.223.1 Detailed Description	3532
6.224find_fn_imps.hpp File Reference	3533
6.224.1 Detailed Description	3533
6.225find_fn_imps.hpp File Reference	3533
6.225.1 Detailed Description	3533
6.226find_fn_imps.hpp File Reference	3533
6.226.1 Detailed Description	3533
6.227find_fn_imps.hpp File Reference	3533
6.227.1 Detailed Description	3533
6.228find_fn_imps.hpp File Reference	3533
6.228.1 Detailed Description	3533
6.229find_fn_imps.hpp File Reference	3533
6.229.1 Detailed Description	3533
6.230find_fn_imps.hpp File Reference	3534
6.230.1 Detailed Description	3534
6.231find_fn_imps.hpp File Reference	3534
6.231.1 Detailed Description	3534
6.232find_fn_imps.hpp File Reference	3534
6.232.1 Detailed Description	3534
6.233find_fn_imps.hpp File Reference	3534
6.233.1 Detailed Description	3534
6.234find_no_store_hash_fn_imps.hpp File Reference	3534
6.234.1 Detailed Description	3534
6.235find_selectors.h File Reference	3534
6.235.1 Detailed Description	3535

6.236find_store_hash_fn_imps.hpp File Reference	3535
6.236.1 Detailed Description	3535
6.237find_store_hash_fn_imps.hpp File Reference	3535
6.237.1 Detailed Description	3535
6.238for_each.h File Reference	3535
6.238.1 Detailed Description	3535
6.239for_each_selectors.h File Reference	3536
6.239.1 Detailed Description	3536
6.240formatter.h File Reference	3536
6.240.1 Detailed Description	3537
6.241forward_list File Reference	3537
6.241.1 Detailed Description	3537
6.242forward_list File Reference	3538
6.242.1 Detailed Description	3538
6.243forward_list File Reference	3539
6.243.1 Detailed Description	3539
6.244forward_list File Reference	3539
6.244.1 Detailed Description	3540
6.245forward_list.h File Reference	3540
6.245.1 Detailed Description	3541
6.246forward_list.tcc File Reference	3541
6.246.1 Detailed Description	3541
6.247fs_dir.h File Reference	3542
6.247.1 Detailed Description	3542
6.248fs_fwd.h File Reference	3542
6.248.1 Detailed Description	3544
6.249fs_path.h File Reference	3544
6.249.1 Detailed Description	3545

6.249.2 Function Documentation	3545
6.250fstream File Reference	3547
6.250.1 Detailed Description	3548
6.251fstream.tcc File Reference	3548
6.251.1 Detailed Description	3548
6.252functexcept.h File Reference	3548
6.252.1 Detailed Description	3549
6.253functional File Reference	3549
6.253.1 Detailed Description	3553
6.254functional File Reference	3553
6.254.1 Detailed Description	3554
6.255functional File Reference	3554
6.255.1 Detailed Description	3555
6.255.2 Function Documentation	3555
6.255.3 Variable Documentation	3556
6.256functional_hash.h File Reference	3556
6.256.1 Detailed Description	3557
6.257functions.h File Reference	3557
6.257.1 Detailed Description	3559
6.258future File Reference	3559
6.258.1 Detailed Description	3560
6.259gp_ht_map_.hpp File Reference	3561
6.259.1 Detailed Description	3561
6.260gslice.h File Reference	3561
6.260.1 Detailed Description	3562
6.261gslice_array.h File Reference	3562
6.261.1 Detailed Description	3562
6.262hash_bytes.h File Reference	3562

6.262.1 Detailed Description	3562
6.263hash_eq_fn.hpp File Reference	3563
6.263.1 Detailed Description	3563
6.264hash_exponential_size_policy_imp.hpp File Reference	3563
6.264.1 Detailed Description	3563
6.265hash_fun.h File Reference	3563
6.265.1 Detailed Description	3563
6.266hash_load_check_resize_trigger_imp.hpp File Reference	3563
6.266.1 Detailed Description	3564
6.267hash_load_check_resize_trigger_size_base.hpp File Reference	3564
6.267.1 Detailed Description	3564
6.268hash_map File Reference	3564
6.268.1 Detailed Description	3565
6.269hash_policy.hpp File Reference	3565
6.269.1 Detailed Description	3566
6.270hash_prime_size_policy_imp.hpp File Reference	3566
6.270.1 Detailed Description	3567
6.271hash_set File Reference	3567
6.271.1 Detailed Description	3567
6.272hash_standard_resize_policy_imp.hpp File Reference	3568
6.272.1 Detailed Description	3568
6.273hashtable.h File Reference	3568
6.273.1 Detailed Description	3568
6.274hashtable.h File Reference	3568
6.274.1 Detailed Description	3569
6.275hashtable_policy.h File Reference	3569
6.275.1 Detailed Description	3571
6.276helper_functions.h File Reference	3571

6.276.1 Detailed Description	3572
6.277indirect_array.h File Reference	3572
6.277.1 Detailed Description	3572
6.278info_fn_imps.hpp File Reference	3572
6.278.1 Detailed Description	3572
6.279info_fn_imps.hpp File Reference	3573
6.279.1 Detailed Description	3573
6.280info_fn_imps.hpp File Reference	3573
6.280.1 Detailed Description	3573
6.281info_fn_imps.hpp File Reference	3573
6.281.1 Detailed Description	3573
6.282info_fn_imps.hpp File Reference	3573
6.282.1 Detailed Description	3573
6.283info_fn_imps.hpp File Reference	3573
6.283.1 Detailed Description	3573
6.284info_fn_imps.hpp File Reference	3573
6.284.1 Detailed Description	3573
6.285info_fn_imps.hpp File Reference	3574
6.285.1 Detailed Description	3574
6.286info_fn_imps.hpp File Reference	3574
6.286.1 Detailed Description	3574
6.287info_fn_imps.hpp File Reference	3574
6.287.1 Detailed Description	3574
6.288initializer_list File Reference	3574
6.288.1 Detailed Description	3574
6.289insert_fn_imps.hpp File Reference	3575
6.289.1 Detailed Description	3575
6.290insert_fn_imps.hpp File Reference	3575

6.290.1 Detailed Description	3575
6.291insert_fn_imps.hpp File Reference	3575
6.291.1 Detailed Description	3575
6.292insert_fn_imps.hpp File Reference	3575
6.292.1 Detailed Description	3575
6.293insert_fn_imps.hpp File Reference	3575
6.293.1 Detailed Description	3575
6.294insert_fn_imps.hpp File Reference	3575
6.294.1 Detailed Description	3575
6.295insert_fn_imps.hpp File Reference	3576
6.295.1 Detailed Description	3576
6.296insert_fn_imps.hpp File Reference	3576
6.296.1 Detailed Description	3576
6.297insert_fn_imps.hpp File Reference	3576
6.297.1 Detailed Description	3576
6.298insert_fn_imps.hpp File Reference	3576
6.298.1 Detailed Description	3576
6.299insert_fn_imps.hpp File Reference	3576
6.299.1 Detailed Description	3576
6.300insert_fn_imps.hpp File Reference	3576
6.300.1 Detailed Description	3576
6.301insert_fn_imps.hpp File Reference	3577
6.301.1 Detailed Description	3577
6.302insert_join_fn_imps.hpp File Reference	3577
6.302.1 Detailed Description	3577
6.303insert_no_store_hash_fn_imps.hpp File Reference	3577
6.303.1 Detailed Description	3577
6.304insert_no_store_hash_fn_imps.hpp File Reference	3577

6.304.1 Detailed Description	3577
6.305insert_store_hash_fn_imps.hpp File Reference	3577
6.305.1 Detailed Description	3577
6.306insert_store_hash_fn_imps.hpp File Reference	3577
6.306.1 Detailed Description	3577
6.307iomanip File Reference	3577
6.307.1 Detailed Description	3579
6.308ios File Reference	3579
6.308.1 Detailed Description	3579
6.309ios_base.h File Reference	3579
6.309.1 Detailed Description	3581
6.310iosfwd File Reference	3581
6.310.1 Detailed Description	3582
6.311iostream File Reference	3582
6.311.1 Detailed Description	3583
6.312istream File Reference	3583
6.312.1 Detailed Description	3584
6.313istream.tcc File Reference	3584
6.313.1 Detailed Description	3585
6.314iterator File Reference	3585
6.314.1 Detailed Description	3585
6.315iterator File Reference	3585
6.315.1 Detailed Description	3585
6.316iterator File Reference	3586
6.316.1 Detailed Description	3586
6.316.2 Function Documentation	3586
6.317iterator.h File Reference	3586
6.317.1 Detailed Description	3587

6.318	iterator.hpp File Reference	3587
6.318.1	Detailed Description	3587
6.319	iterator_fn_imps.hpp File Reference	3587
6.319.1	Detailed Description	3587
6.320	iterator_tracker.h File Reference	3587
6.320.1	Detailed Description	3588
6.321	iterators_fn_imps.hpp File Reference	3589
6.321.1	Detailed Description	3589
6.322	iterators_fn_imps.hpp File Reference	3589
6.322.1	Detailed Description	3589
6.323	iterators_fn_imps.hpp File Reference	3589
6.323.1	Detailed Description	3589
6.324	iterators_fn_imps.hpp File Reference	3589
6.324.1	Detailed Description	3589
6.325	iterators_fn_imps.hpp File Reference	3589
6.325.1	Detailed Description	3589
6.326	iterators_fn_imps.hpp File Reference	3589
6.326.1	Detailed Description	3589
6.327	iterators_fn_imps.hpp File Reference	3590
6.327.1	Detailed Description	3590
6.328	left_child_next_sibling_heap_.hpp File Reference	3590
6.328.1	Detailed Description	3590
6.329	lfts_config.h File Reference	3590
6.329.1	Detailed Description	3590
6.330	limits File Reference	3591
6.330.1	Detailed Description	3592
6.331	linear_probe_fn_imp.hpp File Reference	3592
6.331.1	Detailed Description	3592

6.332list File Reference	3592
6.332.1 Detailed Description	3592
6.333list File Reference	3592
6.333.1 Detailed Description	3593
6.334list File Reference	3593
6.334.1 Detailed Description	3594
6.335list File Reference	3594
6.335.1 Detailed Description	3594
6.336list.tcc File Reference	3595
6.336.1 Detailed Description	3595
6.337list_partition.h File Reference	3595
6.337.1 Detailed Description	3595
6.338list_update_policy.hpp File Reference	3595
6.338.1 Detailed Description	3596
6.339locale File Reference	3596
6.339.1 Detailed Description	3596
6.340locale_classes.h File Reference	3596
6.340.1 Detailed Description	3596
6.341locale_classes.tcc File Reference	3596
6.341.1 Detailed Description	3597
6.342locale_conv.h File Reference	3597
6.342.1 Detailed Description	3598
6.343locale_facets.h File Reference	3598
6.343.1 Detailed Description	3599
6.344locale_facets.tcc File Reference	3599
6.344.1 Detailed Description	3600
6.345locale_facets_nonio.h File Reference	3600
6.345.1 Detailed Description	3600

6.346locale_facets_nonio.tcc File Reference	3601
6.346.1 Detailed Description	3601
6.347localefwd.h File Reference	3601
6.347.1 Detailed Description	3602
6.348losertree.h File Reference	3602
6.348.1 Detailed Description	3603
6.349lu_counter_metadata.hpp File Reference	3603
6.349.1 Detailed Description	3603
6.350lu_map_.hpp File Reference	3603
6.350.1 Detailed Description	3604
6.351macros.h File Reference	3604
6.351.1 Detailed Description	3605
6.351.2 Macro Definition Documentation	3605
6.352malloc_allocator.h File Reference	3607
6.352.1 Detailed Description	3607
6.353map File Reference	3607
6.353.1 Detailed Description	3607
6.354map File Reference	3608
6.354.1 Detailed Description	3608
6.355map File Reference	3608
6.355.1 Detailed Description	3608
6.356map File Reference	3608
6.356.1 Detailed Description	3609
6.357map.h File Reference	3609
6.357.1 Detailed Description	3610
6.358map.h File Reference	3610
6.358.1 Detailed Description	3610
6.359mask_array.h File Reference	3610

6.359.1 Detailed Description	3611
6.360mask_based_range_hashing.hpp File Reference	3611
6.360.1 Detailed Description	3611
6.361math.h File Reference	3611
6.361.1 Detailed Description	3611
6.362memory File Reference	3611
6.362.1 Detailed Description	3612
6.363memory File Reference	3612
6.363.1 Detailed Description	3613
6.364memory File Reference	3613
6.364.1 Detailed Description	3614
6.365memory_resource File Reference	3614
6.365.1 Detailed Description	3615
6.366memoryfwd.h File Reference	3615
6.366.1 Detailed Description	3615
6.367merge.h File Reference	3615
6.367.1 Detailed Description	3616
6.368messages_members.h File Reference	3616
6.368.1 Detailed Description	3616
6.369mod_based_range_hashing.hpp File Reference	3616
6.369.1 Detailed Description	3617
6.370move.h File Reference	3617
6.370.1 Detailed Description	3617
6.371mt_allocator.h File Reference	3618
6.371.1 Detailed Description	3618
6.372multimap.h File Reference	3618
6.372.1 Detailed Description	3619
6.373multimap.h File Reference	3619

6.373.1 Detailed Description	3620
6.374multiseq_selection.h File Reference	3620
6.374.1 Detailed Description	3621
6.375multiset.h File Reference	3621
6.375.1 Detailed Description	3622
6.376multiset.h File Reference	3622
6.376.1 Detailed Description	3622
6.377multiway_merge.h File Reference	3623
6.377.1 Detailed Description	3625
6.377.2 Macro Definition Documentation	3626
6.378multiway_mergesort.h File Reference	3626
6.378.1 Detailed Description	3626
6.379mutex File Reference	3626
6.379.1 Detailed Description	3627
6.380nested_exception.h File Reference	3627
6.380.1 Detailed Description	3628
6.381new File Reference	3628
6.381.1 Detailed Description	3629
6.381.2 Function Documentation	3629
6.382new_allocator.h File Reference	3633
6.382.1 Detailed Description	3633
6.383node.hpp File Reference	3633
6.383.1 Detailed Description	3634
6.384node.hpp File Reference	3634
6.384.1 Detailed Description	3634
6.385node.hpp File Reference	3634
6.385.1 Detailed Description	3634
6.386node_iterators.hpp File Reference	3634

6.386.1 Detailed Description	3635
6.387node_iterators.hpp File Reference	3635
6.387.1 Detailed Description	3635
6.388node_metadata_selector.hpp File Reference	3635
6.388.1 Detailed Description	3636
6.389node_metadata_selector.hpp File Reference	3636
6.389.1 Detailed Description	3636
6.390null_node_metadata.hpp File Reference	3636
6.390.1 Detailed Description	3636
6.391numeric File Reference	3637
6.391.1 Detailed Description	3637
6.392numeric File Reference	3637
6.392.1 Detailed Description	3637
6.393numeric File Reference	3637
6.393.1 Detailed Description	3639
6.394numeric File Reference	3639
6.394.1 Detailed Description	3640
6.395numeric_traits.h File Reference	3640
6.395.1 Detailed Description	3640
6.396numeric_fwd.h File Reference	3641
6.396.1 Detailed Description	3642
6.397omp_loop.h File Reference	3642
6.397.1 Detailed Description	3643
6.398omp_loop_static.h File Reference	3643
6.398.1 Detailed Description	3643
6.399opt_random.h File Reference	3643
6.399.1 Detailed Description	3643
6.400optional File Reference	3643

6.400.1 Detailed Description	3646
6.400.2 Function Documentation	3646
6.401order_statistics_imp.hpp File Reference	3646
6.401.1 Detailed Description	3646
6.402order_statistics_imp.hpp File Reference	3647
6.402.1 Detailed Description	3647
6.403ordered_base.h File Reference	3647
6.403.1 Detailed Description	3647
6.404os_defines.h File Reference	3647
6.404.1 Detailed Description	3647
6.405ostream File Reference	3647
6.405.1 Detailed Description	3648
6.406ostream.tcc File Reference	3649
6.406.1 Detailed Description	3649
6.407ostream_insert.h File Reference	3649
6.407.1 Detailed Description	3649
6.408ov_tree_map_.hpp File Reference	3650
6.408.1 Detailed Description	3650
6.409pairing_heap_.hpp File Reference	3650
6.409.1 Detailed Description	3650
6.410par_loop.h File Reference	3651
6.410.1 Detailed Description	3651
6.411parallel.h File Reference	3651
6.411.1 Detailed Description	3651
6.412parse_numbers.h File Reference	3651
6.412.1 Detailed Description	3651
6.413partial_sum.h File Reference	3652
6.413.1 Detailed Description	3652

6.414partition.h File Reference	3652
6.414.1 Detailed Description	3653
6.414.2 Macro Definition Documentation	3653
6.415pat_trie.hpp File Reference	3653
6.415.1 Detailed Description	3653
6.416pat_trie_base.hpp File Reference	3654
6.416.1 Detailed Description	3654
6.417pod_char_traits.h File Reference	3654
6.417.1 Detailed Description	3655
6.418point_const_iterator.hpp File Reference	3655
6.418.1 Detailed Description	3655
6.419point_const_iterator.hpp File Reference	3655
6.419.1 Detailed Description	3656
6.420point_const_iterator.hpp File Reference	3656
6.420.1 Detailed Description	3656
6.421point_iterator.hpp File Reference	3656
6.421.1 Detailed Description	3656
6.422point_iterators.hpp File Reference	3656
6.422.1 Detailed Description	3657
6.423pointer.h File Reference	3657
6.423.1 Detailed Description	3659
6.424policy_access_fn_imps.hpp File Reference	3659
6.424.1 Detailed Description	3659
6.425policy_access_fn_imps.hpp File Reference	3659
6.425.1 Detailed Description	3659
6.426policy_access_fn_imps.hpp File Reference	3659
6.426.1 Detailed Description	3659
6.427policy_access_fn_imps.hpp File Reference	3659

6.427.1 Detailed Description	3659
6.428policy_access_fn_imps.hpp File Reference	3659
6.428.1 Detailed Description	3659
6.429policy_access_fn_imps.hpp File Reference	3660
6.429.1 Detailed Description	3660
6.430policy_access_fn_imps.hpp File Reference	3660
6.430.1 Detailed Description	3660
6.431pool_allocator.h File Reference	3660
6.431.1 Detailed Description	3660
6.432postypes.h File Reference	3660
6.432.1 Detailed Description	3661
6.433predefined_ops.h File Reference	3661
6.433.1 Detailed Description	3662
6.434prefix_search_node_update_imp.hpp File Reference	3662
6.434.1 Detailed Description	3662
6.435priority_queue.hpp File Reference	3662
6.435.1 Detailed Description	3662
6.436priority_queue_base_dispatch.hpp File Reference	3663
6.436.1 Detailed Description	3663
6.437probe_fn_base.hpp File Reference	3663
6.437.1 Detailed Description	3663
6.438profiler.h File Reference	3663
6.438.1 Detailed Description	3665
6.439profiler_algos.h File Reference	3665
6.439.1 Detailed Description	3666
6.440profiler_container_size.h File Reference	3666
6.440.1 Detailed Description	3666
6.441profiler_hash_func.h File Reference	3666

6.441.1 Detailed Description	3667
6.442profiler_hashtable_size.h File Reference	3667
6.442.1 Detailed Description	3667
6.443profiler_list_to_slist.h File Reference	3667
6.443.1 Detailed Description	3668
6.444profiler_list_to_vector.h File Reference	3668
6.444.1 Detailed Description	3668
6.445profiler_map_to_unordered_map.h File Reference	3669
6.445.1 Detailed Description	3669
6.446profiler_node.h File Reference	3669
6.446.1 Detailed Description	3670
6.447profiler_state.h File Reference	3670
6.447.1 Detailed Description	3670
6.448profiler_trace.h File Reference	3671
6.448.1 Detailed Description	3673
6.449profiler_vector_size.h File Reference	3673
6.449.1 Detailed Description	3673
6.450profiler_vector_to_list.h File Reference	3673
6.450.1 Detailed Description	3674
6.451propagate_const File Reference	3674
6.451.1 Detailed Description	3676
6.452ptr_traits.h File Reference	3676
6.452.1 Detailed Description	3677
6.453quadratic_probe_fn_imp.hpp File Reference	3677
6.453.1 Detailed Description	3677
6.454queue File Reference	3677
6.454.1 Detailed Description	3677
6.455queue.h File Reference	3677

6.455.1 Detailed Description	3677
6.455.2 Macro Definition Documentation	3678
6.456quicksort.h File Reference	3678
6.456.1 Detailed Description	3678
6.457quoted_string.h File Reference	3678
6.457.1 Detailed Description	3679
6.458r_erase_fn_imps.hpp File Reference	3679
6.458.1 Detailed Description	3679
6.459r_erase_fn_imps.hpp File Reference	3679
6.459.1 Detailed Description	3679
6.460random File Reference	3679
6.460.1 Detailed Description	3679
6.461random File Reference	3679
6.461.1 Detailed Description	3680
6.462random.h File Reference	3680
6.462.1 Detailed Description	3684
6.463random.tcc File Reference	3684
6.463.1 Detailed Description	3688
6.464random.tcc File Reference	3688
6.464.1 Detailed Description	3690
6.465random_number.h File Reference	3690
6.465.1 Detailed Description	3690
6.466random_shuffle.h File Reference	3690
6.466.1 Detailed Description	3691
6.467range_access.h File Reference	3691
6.467.1 Detailed Description	3692
6.468ranged_hash_fn.hpp File Reference	3693
6.468.1 Detailed Description	3693

6.469	ranged_probe_fn.hpp File Reference	3693
6.469.1	Detailed Description	3694
6.470	ratio File Reference	3694
6.470.1	Detailed Description	3694
6.471	ratio File Reference	3694
6.471.1	Detailed Description	3695
6.472	ratio File Reference	3695
6.472.1	Detailed Description	3695
6.473	rb_tree File Reference	3695
6.473.1	Detailed Description	3696
6.474	rb_tree.hpp File Reference	3696
6.474.1	Detailed Description	3696
6.475	rc.hpp File Reference	3696
6.475.1	Detailed Description	3697
6.476	rc_binomial_heap.hpp File Reference	3697
6.476.1	Detailed Description	3697
6.477	rc_string_base.h File Reference	3697
6.477.1	Detailed Description	3697
6.478	regex File Reference	3698
6.478.1	Detailed Description	3698
6.479	regex File Reference	3698
6.479.1	Detailed Description	3698
6.480	regex.h File Reference	3698
6.480.1	Detailed Description	3703
6.481	regex.tcc File Reference	3703
6.481.1	Detailed Description	3704
6.482	regex_automaton.h File Reference	3704
6.482.1	Detailed Description	3705

6.483regex_automaton.tcc File Reference	3705
6.483.1 Detailed Description	3705
6.484regex_compiler.h File Reference	3705
6.484.1 Detailed Description	3706
6.485regex_compiler.tcc File Reference	3706
6.485.1 Detailed Description	3706
6.486regex_constants.h File Reference	3706
6.486.1 Detailed Description	3708
6.487regex_error.h File Reference	3708
6.487.1 Detailed Description	3709
6.488regex_executor.h File Reference	3709
6.488.1 Detailed Description	3709
6.489regex_executor.tcc File Reference	3709
6.489.1 Detailed Description	3709
6.490regex_scanner.h File Reference	3709
6.490.1 Detailed Description	3710
6.491regex_scanner.tcc File Reference	3710
6.491.1 Detailed Description	3710
6.492resize_fn_imps.hpp File Reference	3710
6.492.1 Detailed Description	3710
6.493resize_fn_imps.hpp File Reference	3710
6.493.1 Detailed Description	3710
6.494resize_no_store_hash_fn_imps.hpp File Reference	3710
6.494.1 Detailed Description	3710
6.495resize_no_store_hash_fn_imps.hpp File Reference	3711
6.495.1 Detailed Description	3711
6.496resize_policy.hpp File Reference	3711
6.496.1 Detailed Description	3711

6.497resize_store_hash_fn_imps.hpp File Reference	3711
6.497.1 Detailed Description	3711
6.498resize_store_hash_fn_imps.hpp File Reference	3711
6.498.1 Detailed Description	3711
6.499rope File Reference	3711
6.499.1 Detailed Description	3715
6.500ropeimpl.h File Reference	3715
6.500.1 Detailed Description	3715
6.501rotate_fn_imps.hpp File Reference	3715
6.501.1 Detailed Description	3715
6.502rotate_fn_imps.hpp File Reference	3715
6.502.1 Detailed Description	3715
6.503safe_base.h File Reference	3716
6.503.1 Detailed Description	3716
6.504safe_container.h File Reference	3716
6.504.1 Detailed Description	3716
6.505safe_iterator.h File Reference	3716
6.505.1 Detailed Description	3718
6.506safe_iterator.tcc File Reference	3718
6.506.1 Detailed Description	3718
6.507safe_local_iterator.h File Reference	3719
6.507.1 Detailed Description	3719
6.508safe_local_iterator.tcc File Reference	3719
6.508.1 Detailed Description	3720
6.509safe_sequence.h File Reference	3720
6.509.1 Detailed Description	3720
6.510safe_sequence.tcc File Reference	3720
6.510.1 Detailed Description	3720

6.511safe_unordered_base.h File Reference	3721
6.511.1 Detailed Description	3721
6.512safe_unordered_container.h File Reference	3721
6.512.1 Detailed Description	3721
6.513safe_unordered_container.tcc File Reference	3721
6.513.1 Detailed Description	3721
6.514sample_probe_fn.hpp File Reference	3722
6.514.1 Detailed Description	3722
6.515sample_range_hashing.hpp File Reference	3722
6.515.1 Detailed Description	3722
6.516sample_ranged_hash_fn.hpp File Reference	3722
6.516.1 Detailed Description	3722
6.517sample_ranged_probe_fn.hpp File Reference	3723
6.517.1 Detailed Description	3723
6.518sample_resize_policy.hpp File Reference	3723
6.518.1 Detailed Description	3723
6.519sample_resize_trigger.hpp File Reference	3723
6.519.1 Detailed Description	3723
6.520sample_size_policy.hpp File Reference	3724
6.520.1 Detailed Description	3724
6.521sample_tree_node_update.hpp File Reference	3724
6.521.1 Detailed Description	3724
6.522sample_trie_access_traits.hpp File Reference	3724
6.522.1 Detailed Description	3724
6.523sample_trie_node_update.hpp File Reference	3725
6.523.1 Detailed Description	3725
6.524sample_update_policy.hpp File Reference	3725
6.524.1 Detailed Description	3725

6.525scoped_allocator File Reference	3725
6.525.1 Detailed Description	3726
6.526search.h File Reference	3726
6.526.1 Detailed Description	3726
6.527set File Reference	3726
6.527.1 Detailed Description	3727
6.528set File Reference	3727
6.528.1 Detailed Description	3727
6.529set File Reference	3727
6.529.1 Detailed Description	3727
6.530set File Reference	3727
6.530.1 Detailed Description	3728
6.531set.h File Reference	3728
6.531.1 Detailed Description	3729
6.532set.h File Reference	3729
6.532.1 Detailed Description	3729
6.533set_operations.h File Reference	3729
6.533.1 Detailed Description	3730
6.534settings.h File Reference	3730
6.534.1 Detailed Description	3730
6.534.2 parallelization_decision	3731
6.534.3 Macro Definition Documentation	3731
6.535shared_mutex File Reference	3732
6.535.1 Detailed Description	3732
6.536shared_ptr.h File Reference	3732
6.536.1 Detailed Description	3734
6.537shared_ptr.h File Reference	3734
6.537.1 Detailed Description	3737

6.537.2 Function Documentation	3737
6.538shared_ptr_atomic.h File Reference	3737
6.538.1 Detailed Description	3738
6.539shared_ptr_base.h File Reference	3738
6.539.1 Detailed Description	3740
6.540size_fn_imps.hpp File Reference	3740
6.540.1 Detailed Description	3740
6.541slice_array.h File Reference	3740
6.541.1 Detailed Description	3741
6.542slist File Reference	3741
6.542.1 Detailed Description	3742
6.543sort.h File Reference	3742
6.543.1 Detailed Description	3742
6.544specfun.h File Reference	3742
6.544.1 Detailed Description	3745
6.545splay_fn_imps.hpp File Reference	3745
6.545.1 Detailed Description	3745
6.546splay_tree_.hpp File Reference	3745
6.546.1 Detailed Description	3745
6.547split_fn_imps.hpp File Reference	3745
6.547.1 Detailed Description	3745
6.548split_join_fn_imps.hpp File Reference	3746
6.548.1 Detailed Description	3746
6.549split_join_fn_imps.hpp File Reference	3746
6.549.1 Detailed Description	3746
6.550split_join_fn_imps.hpp File Reference	3746
6.550.1 Detailed Description	3746
6.551split_join_fn_imps.hpp File Reference	3746

6.551.1 Detailed Description	3746
6.552split_join_fn_imps.hpp File Reference	3746
6.552.1 Detailed Description	3746
6.553split_join_fn_imps.hpp File Reference	3746
6.553.1 Detailed Description	3746
6.554split_join_fn_imps.hpp File Reference	3747
6.554.1 Detailed Description	3747
6.555split_join_fn_imps.hpp File Reference	3747
6.555.1 Detailed Description	3747
6.556split_join_fn_imps.hpp File Reference	3747
6.556.1 Detailed Description	3747
6.557sso_string_base.h File Reference	3747
6.557.1 Detailed Description	3747
6.558sstream File Reference	3747
6.558.1 Detailed Description	3748
6.559sstream.tcc File Reference	3748
6.559.1 Detailed Description	3748
6.560stack File Reference	3748
6.560.1 Detailed Description	3749
6.561standard_policies.hpp File Reference	3749
6.561.1 Detailed Description	3749
6.561.2 Enumeration Type Documentation	3749
6.562std_mutex.h File Reference	3750
6.562.1 Detailed Description	3750
6.563stdc++.h File Reference	3750
6.563.1 Detailed Description	3750
6.564stdexcept File Reference	3751
6.564.1 Detailed Description	3751

6.565	stdio_filebuf.h File Reference	3751
6.565.1	Detailed Description	3751
6.566	stdio_sync_filebuf.h File Reference	3752
6.566.1	Detailed Description	3752
6.567	stdlib.h File Reference	3752
6.567.1	Detailed Description	3752
6.568	stdtr1c++.h File Reference	3752
6.568.1	Detailed Description	3752
6.569	stl_algo.h File Reference	3752
6.569.1	Detailed Description	3762
6.569.2	Function Documentation	3762
6.570	stl_algobase.h File Reference	3763
6.570.1	Detailed Description	3765
6.571	stl_bvector.h File Reference	3765
6.571.1	Detailed Description	3766
6.572	stl_construct.h File Reference	3766
6.572.1	Detailed Description	3767
6.573	stl_deque.h File Reference	3767
6.573.1	Detailed Description	3769
6.573.2	Macro Definition Documentation	3769
6.574	stl_function.h File Reference	3769
6.574.1	Detailed Description	3771
6.575	stl_heap.h File Reference	3772
6.575.1	Detailed Description	3773
6.576	stl_iterator.h File Reference	3773
6.576.1	Detailed Description	3776
6.577	stl_iterator.h File Reference	3776
6.577.1	Detailed Description	3777

6.578	stl_iterator_base_funcs.h File Reference	3777
6.578.1	Detailed Description	3778
6.579	stl_iterator_base_types.h File Reference	3778
6.579.1	Detailed Description	3779
6.580	stl_list.h File Reference	3779
6.580.1	Detailed Description	3780
6.581	stl_map.h File Reference	3780
6.581.1	Detailed Description	3780
6.582	stl_multimap.h File Reference	3781
6.582.1	Detailed Description	3781
6.583	stl_multiset.h File Reference	3781
6.583.1	Detailed Description	3782
6.584	stl_numeric.h File Reference	3782
6.584.1	Detailed Description	3783
6.585	stl_pair.h File Reference	3783
6.585.1	Detailed Description	3784
6.586	stl_queue.h File Reference	3784
6.586.1	Detailed Description	3785
6.587	stl_raw_storage_iter.h File Reference	3785
6.587.1	Detailed Description	3785
6.588	stl_relops.h File Reference	3785
6.588.1	Detailed Description	3786
6.589	stl_set.h File Reference	3786
6.589.1	Detailed Description	3787
6.590	stl_stack.h File Reference	3787
6.590.1	Detailed Description	3787
6.591	stl_tempbuf.h File Reference	3787
6.591.1	Detailed Description	3788

6.592stl_tree.h File Reference	3788
6.592.1 Detailed Description	3789
6.593stl_uninitialized.h File Reference	3789
6.593.1 Detailed Description	3791
6.594stl_vector.h File Reference	3791
6.594.1 Detailed Description	3792
6.595stream_iterator.h File Reference	3792
6.595.1 Detailed Description	3792
6.596streambuf File Reference	3792
6.596.1 Detailed Description	3793
6.597streambuf.tcc File Reference	3793
6.597.1 Detailed Description	3793
6.598streambuf_iterator.h File Reference	3793
6.598.1 Detailed Description	3794
6.599string File Reference	3794
6.599.1 Detailed Description	3794
6.600string File Reference	3795
6.600.1 Detailed Description	3797
6.601string File Reference	3797
6.601.1 Detailed Description	3797
6.602string_conversions.h File Reference	3798
6.602.1 Detailed Description	3798
6.603string_view File Reference	3798
6.603.1 Detailed Description	3800
6.604string_view.tcc File Reference	3800
6.604.1 Detailed Description	3800
6.605stringfwd.h File Reference	3800
6.605.1 Detailed Description	3801

6.606	sstream File Reference	3801
6.606.1	Detailed Description	3801
6.607	synth_access_traits.hpp File Reference	3801
6.607.1	Detailed Description	3801
6.608	system_error File Reference	3802
6.608.1	Detailed Description	3803
6.609	system_error File Reference	3803
6.609.1	Detailed Description	3803
6.610	tag_and_trait.hpp File Reference	3803
6.610.1	Detailed Description	3804
6.611	tags.h File Reference	3805
6.611.1	Detailed Description	3805
6.612	tgmath.h File Reference	3805
6.612.1	Detailed Description	3805
6.613	thin_heap_.hpp File Reference	3805
6.613.1	Detailed Description	3806
6.614	thread File Reference	3806
6.614.1	Detailed Description	3807
6.615	throw_allocator.h File Reference	3807
6.615.1	Detailed Description	3808
6.616	time_members.h File Reference	3808
6.616.1	Detailed Description	3809
6.617	trace_fn_imps.hpp File Reference	3809
6.617.1	Detailed Description	3809
6.618	trace_fn_imps.hpp File Reference	3809
6.618.1	Detailed Description	3809
6.619	trace_fn_imps.hpp File Reference	3809
6.619.1	Detailed Description	3809

6.620trace_fn_imps.hpp File Reference	3809
6.620.1 Detailed Description	3809
6.621trace_fn_imps.hpp File Reference	3809
6.621.1 Detailed Description	3809
6.622trace_fn_imps.hpp File Reference	3809
6.622.1 Detailed Description	3809
6.623trace_fn_imps.hpp File Reference	3810
6.623.1 Detailed Description	3810
6.624trace_fn_imps.hpp File Reference	3810
6.624.1 Detailed Description	3810
6.625traits.hpp File Reference	3810
6.625.1 Detailed Description	3810
6.626traits.hpp File Reference	3810
6.626.1 Detailed Description	3811
6.627traits.hpp File Reference	3811
6.627.1 Detailed Description	3811
6.628traits.hpp File Reference	3811
6.628.1 Detailed Description	3811
6.629traits.hpp File Reference	3811
6.629.1 Detailed Description	3812
6.630traits.hpp File Reference	3812
6.630.1 Detailed Description	3812
6.631tree_policy.hpp File Reference	3812
6.631.1 Detailed Description	3812
6.632tree_trace_base.hpp File Reference	3813
6.632.1 Detailed Description	3813
6.633trie_policy.hpp File Reference	3813
6.633.1 Detailed Description	3813

6.634trie_policy_base.hpp File Reference	3813
6.634.1 Detailed Description	3814
6.635trie_string_access_traits_imp.hpp File Reference	3814
6.635.1 Detailed Description	3814
6.636tuple File Reference	3814
6.636.1 Detailed Description	3816
6.637tuple File Reference	3816
6.637.1 Detailed Description	3816
6.638type_traits File Reference	3817
6.638.1 Detailed Description	3818
6.639type_traits File Reference	3818
6.639.1 Detailed Description	3818
6.640type_traits File Reference	3819
6.640.1 Detailed Description	3822
6.641type_traits.h File Reference	3822
6.641.1 Detailed Description	3822
6.642type_utils.hpp File Reference	3823
6.642.1 Detailed Description	3823
6.643typeid File Reference	3823
6.643.1 Detailed Description	3823
6.644typeid File Reference	3824
6.644.1 Detailed Description	3824
6.645typelist.h File Reference	3824
6.645.1 Detailed Description	3825
6.646types.h File Reference	3825
6.646.1 Detailed Description	3826
6.647types_traits.hpp File Reference	3826
6.647.1 Detailed Description	3827

6.648uniform_int_dist.h File Reference	3827
6.648.1 Detailed Description	3827
6.649unique_copy.h File Reference	3827
6.649.1 Detailed Description	3827
6.650unique_ptr.h File Reference	3828
6.650.1 Detailed Description	3829
6.651unordered_base.h File Reference	3829
6.651.1 Detailed Description	3829
6.652unordered_map File Reference	3829
6.652.1 Detailed Description	3830
6.653unordered_map File Reference	3830
6.653.1 Detailed Description	3830
6.654unordered_map File Reference	3831
6.654.1 Detailed Description	3831
6.655unordered_map File Reference	3832
6.655.1 Detailed Description	3832
6.656unordered_map.h File Reference	3832
6.656.1 Detailed Description	3833
6.657unordered_set File Reference	3833
6.657.1 Detailed Description	3834
6.658unordered_set File Reference	3834
6.658.1 Detailed Description	3834
6.659unordered_set File Reference	3835
6.659.1 Detailed Description	3835
6.660unordered_set File Reference	3836
6.660.1 Detailed Description	3836
6.661unordered_set.h File Reference	3836
6.661.1 Detailed Description	3837

6.662update_fn_imps.hpp File Reference	3837
6.662.1 Detailed Description	3837
6.663utility File Reference	3838
6.663.1 Detailed Description	3839
6.664utility File Reference	3839
6.664.1 Detailed Description	3839
6.665valarray File Reference	3840
6.665.1 Detailed Description	3843
6.666valarray_after.h File Reference	3843
6.666.1 Detailed Description	3853
6.667valarray_array.h File Reference	3853
6.667.1 Detailed Description	3861
6.668valarray_array.tcc File Reference	3861
6.668.1 Detailed Description	3862
6.669valarray_before.h File Reference	3862
6.669.1 Detailed Description	3862
6.670vector File Reference	3863
6.670.1 Detailed Description	3863
6.671vector File Reference	3863
6.671.1 Detailed Description	3864
6.672vector File Reference	3864
6.672.1 Detailed Description	3864
6.673vector File Reference	3864
6.673.1 Detailed Description	3865
6.674vector.tcc File Reference	3865
6.674.1 Detailed Description	3865
6.675vstring.h File Reference	3865
6.675.1 Detailed Description	3868
6.676vstring.tcc File Reference	3868
6.676.1 Detailed Description	3869
6.677vstring_fwd.h File Reference	3869
6.677.1 Detailed Description	3869
6.678vstring_util.h File Reference	3869
6.678.1 Detailed Description	3870
6.679workstealing.h File Reference	3870
6.679.1 Detailed Description	3870

1 Mathematical Special Functions

1.1 Introduction and History

The first significant library upgrade on the road to C++2011, [TR1](#), included a set of 23 mathematical functions that significantly extended the standard transcendental functions inherited from C and declared in `<cmath>`.

Although most components from TR1 were eventually adopted for C++11 these math functions were left behind out of concern for implementability. The math functions were published as a separate international standard [IS 29124 - Extensions to the C++ Library to Support Mathematical Special Functions](#).

For C++17 these functions were incorporated into the main standard.

1.2 Contents

The following functions are implemented in namespace `std`:

- [assoc_laguerre](#) - Associated Laguerre functions
- [assoc_legendre](#) - Associated Legendre functions
- [beta](#) - Beta functions
- [comp_ellint_1](#) - Complete elliptic functions of the first kind
- [comp_ellint_2](#) - Complete elliptic functions of the second kind
- [comp_ellint_3](#) - Complete elliptic functions of the third kind
- [cyl_bessel_i](#) - Regular modified cylindrical Bessel functions
- [cyl_bessel_j](#) - Cylindrical Bessel functions of the first kind
- [cyl_bessel_k](#) - Irregular modified cylindrical Bessel functions
- [cyl_neumann](#) - Cylindrical Neumann functions or Cylindrical Bessel functions of the second kind
- [ellint_1](#) - Incomplete elliptic functions of the first kind
- [ellint_2](#) - Incomplete elliptic functions of the second kind
- [ellint_3](#) - Incomplete elliptic functions of the third kind
- [expint](#) - The exponential integral
- [hermite](#) - Hermite polynomials
- [laguerre](#) - Laguerre functions
- [legendre](#) - Legendre polynomials
- [riemann_zeta](#) - The Riemann zeta function

- [sph_bessel](#) - Spherical Bessel functions
- [sph_legendre](#) - Spherical Legendre functions
- [sph_neumann](#) - Spherical Neumann functions

The hypergeometric functions were stricken from the TR29124 and C++17 versions of this math library because of implementation concerns. However, since they were in the TR1 version and since they are popular we kept them as an extension in namespace `__gnu_cxx`:

- `conf_hyperg` - Confluent hypergeometric functions
- `hyperg` - Hypergeometric functions

1.3 General Features

1.3.1 Argument Promotion

The arguments supplied to the non-suffixed functions will be promoted according to the following rules: 1. If any argument intended to be floating point is given an integral value That integral value is promoted to double. 2. All floating point arguments are promoted up to the largest floating point precision among them.

1.3.2 NaN Arguments

If any of the floating point arguments supplied to these functions is invalid or NaN (`std::numeric_limits<Tp>::quiet_NaN`), the value NaN is returned.

1.4 Implementation

We strive to implement the underlying math with type generic algorithms to the greatest extent possible. In practice, the functions are thin wrappers that dispatch to function templates. Type dependence is controlled with `std::numeric_limits` and functions thereof.

We don't promote `float` to `double` or `double` to `long double` reflexively. The goal is for `float` functions to operate more quickly, at the cost of `float` accuracy and possibly a smaller domain of validity. Similarly, `long double` should give you more dynamic range and slightly more precision than `double` on many systems.

1.5 Testing

These functions have been tested against equivalent implementations from the [Gnu Scientific Library](http://www.gnu.org/software/scientific/), [GSL](http://www.boost.org/doc/libs/1_60_0/libs/math/doc/html/index.html) and [Boost](http://www.boost.org/doc/libs/1_60_0/libs/math/doc/html/index.html) and the ratio

$$\frac{|f - f_{test}|}{|f_{test}|}$$

is generally found to be within 10^{-15} for 64-bit double on linux-x86_64 systems over most of the ranges of validity.

Todo Provide accuracy comparisons on a per-function basis for a small number of targets.

1.6 General Bibliography

See also

Abramowitz and Stegun: Handbook of Mathematical Functions, with Formulas, Graphs, and Mathematical Tables
 Edited by Milton Abramowitz and Irene A. Stegun, National Bureau of Standards Applied Mathematics Series - 55
 Issued June 1964, Tenth Printing, December 1972, with corrections Electronic versions of A&S abound including
 both pdf and navigable html.

for example <http://people.math.sfu.ca/~cbm/aands/>

The old A&S has been redone as the NIST Digital Library of Mathematical Functions: <http://dlmf.nist.gov/> This version is far more navigable and includes more recent work.

An Atlas of Functions: with Equator, the Atlas Function Calculator 2nd Edition, by Oldham, Keith B., Myland, Jan, Spanier, Jerome

Asymptotics and Special Functions by Frank W. J. Olver, Academic Press, 1974

Numerical Recipes in C, The Art of Scientific Computing, by William H. Press, Second Ed., Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery, Cambridge University Press, 1992

The Special Functions and Their Approximations: Volumes 1 and 2, by Yudell L. Luke, Academic Press, 1969

2 Todo List

Member `__gnu_cxx::distance` (`_InputIterator __first`, `_InputIterator __last`, `_Distance &__n`)

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation__style.html

Class `__gnu_cxx::hash_map<_Key, _Tp, _HashFn, _EqualKey, _Alloc>`

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation__style.html

Class `__gnu_cxx::hash_multimap<_Key, _Tp, _HashFn, _EqualKey, _Alloc>`

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation__style.html

Class `__gnu_cxx::hash_multiset<_Value, _HashFcn, _EqualKey, _Alloc>`

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation__style.html

Class `__gnu_cxx::hash_set<_Value, _HashFcn, _EqualKey, _Alloc>`

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation__style.html

Member `__gnu_cxx::power` (`_Tp __x`, `_Integer __n`)

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation__style.html

Member `__gnu_cxx::power` (`_Tp __x`, `_Integer __n`, `_MonoidOperation __monoid_op`)

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation__style.html

Member `__gnu_cxx::random_sample` (`_InputIterator __first`, `_InputIterator __last`, `_RandomAccessIterator __out_first`, `_RandomAccessIterator __out_last`, `_RandomNumberGenerator &__rand`)

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation__style.html

Member [__gnu_cxx::random_sample](#) ([_InputIterator __first](#), [_InputIterator __last](#), [_RandomAccessIterator __out_first](#), [_RandomAccessIterator __out_last](#))

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

Member [__gnu_cxx::random_sample_n](#) ([_ForwardIterator __first](#), [_ForwardIterator __last](#), [_OutputIterator __out](#), [const _Distance __n](#))

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

Member [__gnu_cxx::random_sample_n](#) ([_ForwardIterator __first](#), [_ForwardIterator __last](#), [_OutputIterator __out](#), [const _Distance __n](#), [_RandomNumberGenerator &__rand](#))

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

Class [__gnu_cxx::rb_tree](#)< [_Key](#), [_Value](#), [_KeyOfValue](#), [_Compare](#), [_Alloc](#) >

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

Class [__gnu_cxx::rope](#)< [_CharT](#), [_Alloc](#) >

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

Class [__gnu_cxx::slist](#)< [_Tp](#), [_Alloc](#) >

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

Member [__gnu_debug::Safe_iterator](#)< [_Iterator](#), [_Sequence](#) >::operator-> () const noexcept

Make this correct w.r.t. iterators that return proxies

Member [__gnu_debug::Safe_local_iterator](#)< [_Iterator](#), [_Sequence](#) >::operator-> () const

Make this correct w.r.t. iterators that return proxies

page [Mathematical Special Functions](#)

Provide accuracy comparisons on a per-function basis for a small number of targets.

Class [std::basic_string](#)< [_CharT](#), [_Traits](#), [_Alloc](#) >

Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

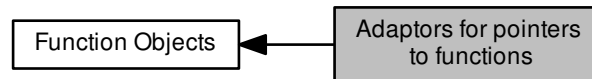
Member [std::regex_traits](#)< [_Ch_type](#) >::transform_primary ([_Fwd_iter __first](#), [_Fwd_iter __last](#)) const

Implement this function correctly.

3 Module Documentation

3.1 Adaptors for pointers to functions

Collaboration diagram for Adaptors for pointers to functions:



Classes

- class `std::pointer_to_binary_function< _Arg1, _Arg2, _Result >`
- class `std::pointer_to_unary_function< _Arg, _Result >`

Functions

- template<typename _Arg, typename _Result >
`pointer_to_unary_function< _Arg, _Result > std::ptr_fun (_Result(*)(_Arg))`
- template<typename _Arg1, typename _Arg2, typename _Result >
`pointer_to_binary_function< _Arg1, _Arg2, _Result > std::ptr_fun (_Result(*)(_Arg1, _Arg2))`

3.1.1 Detailed Description

The advantage of function objects over pointers to functions is that the objects in the standard library declare nested typedefs describing their argument and result types with uniform names (e.g., `result_type` from the base classes `unary_function` and `binary_function`). Sometimes those typedefs are required, not just optional.

Adaptors are provided to turn pointers to unary (single-argument) and binary (double-argument) functions into function objects. The long-winded functor `pointer_to_unary_function` is constructed with a function pointer `f`, and its `operator()` called with argument `x` returns `f(x)`. The functor `pointer_to_binary_function` does the same thing, but with a double-argument `f` and `operator()`.

The function `ptr_fun` takes a pointer-to-function `f` and constructs an instance of the appropriate functor.

3.1.2 Function Documentation

3.1.2.1 `template<typename _Arg, typename _Result > pointer_to_unary_function<_Arg, _Result> std::ptr_fun (_Result(*)(_Arg) __x) [inline]`

One of the [adaptors for function pointers](#).

Definition at line 838 of file `stl_function.h`.

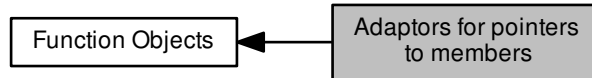
3.1.2.2 `template<typename _Arg1, typename _Arg2, typename _Result > pointer_to_binary_function<_Arg1, _Arg2, _Result> std::ptr_fun (_Result(*)(_Arg1, _Arg2) __x) [inline]`

One of the [adaptors for function pointers](#).

Definition at line 864 of file `stl_function.h`.

3.2 Adaptors for pointers to members

Collaboration diagram for Adaptors for pointers to members:



Classes

- class `std::const_mem_fun1_ref_t<_Ret, _Tp, _Arg>`
- class `std::const_mem_fun1_t<_Ret, _Tp, _Arg>`
- class `std::const_mem_fun_ref_t<_Ret, _Tp>`
- class `std::const_mem_fun_t<_Ret, _Tp>`
- class `std::mem_fun1_ref_t<_Ret, _Tp, _Arg>`
- class `std::mem_fun1_t<_Ret, _Tp, _Arg>`
- class `std::mem_fun_ref_t<_Ret, _Tp>`
- class `std::mem_fun_t<_Ret, _Tp>`

Functions

- `template<typename _Ret, typename _Tp>`
`mem_fun_t<_Ret, _Tp> std::mem_fun (_Ret(_Tp::*__f)())`
- `template<typename _Ret, typename _Tp, typename _Arg>`
`mem_fun1_t<_Ret, _Tp, _Arg> std::mem_fun (_Ret(_Tp::*__f)(_Arg))`
- `template<typename _Ret, typename _Tp>`
`mem_fun_ref_t<_Ret, _Tp> std::mem_fun_ref (_Ret(_Tp::*__f)())`
- `template<typename _Ret, typename _Tp, typename _Arg>`
`mem_fun1_ref_t<_Ret, _Tp, _Arg> std::mem_fun_ref (_Ret(_Tp::*__f)(_Arg))`

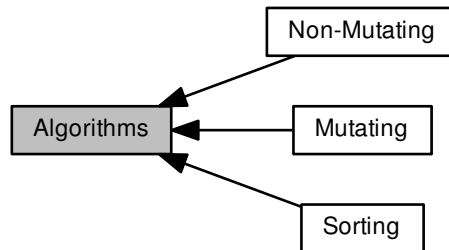
3.2.1 Detailed Description

There are a total of $8 = 2^3$ function objects in this family. (1) Member functions taking no arguments vs member functions taking one argument. (2) Call through pointer vs call through reference. (3) Const vs non-const member function.

All of this complexity is in the function objects themselves. You can ignore it by using the helper function `mem_fun` and `mem_fun_ref`, which create whichever type of adaptor is appropriate.

3.3 Algorithms

Collaboration diagram for Algorithms:



Modules

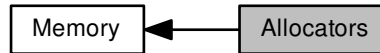
- [Mutating](#)
- [Non-Mutating](#)
- [Sorting](#)

3.3.1 Detailed Description

Components for performing algorithmic operations. Includes non-modifying sequence, modifying (mutating) sequence, sorting, searching, merge, partition, heap, set, minima, maxima, and permutation operations.

3.4 Allocators

Collaboration diagram for Allocators:



Classes

- struct `__gnu_cxx::__alloc_traits<_Alloc>`
- class `__gnu_cxx::__mt_alloc<_Tp, _Poolp>`
- class `__gnu_cxx::__pool_alloc<_Tp>`
- class `__gnu_cxx::__ExtPtr_allocator<_Tp>`
- class `__gnu_cxx::array_allocator<_Tp, _Array>`
- class `__gnu_cxx::bitmap_allocator<_Tp>`
- class `__gnu_cxx::debug_allocator<_Alloc>`
- class `__gnu_cxx::malloc_allocator<_Tp>`
- class `__gnu_cxx::new_allocator<_Tp>`
- class `__gnu_cxx::throw_allocator_base<_Tp, _Cond>`
- class `std::allocator<_Tp>`
- class `std::allocator<void>`
- struct `std::allocator_traits<_Alloc>`
- class `std::scoped_allocator_adaptor<_OuterAlloc, _InnerAllocs>`
- struct `std::uses_allocator<_Tp, _Alloc>`

Typedefs

- template<typename `_Tp`>
using `std::__allocator_base` = `__gnu_cxx::new_allocator<_Tp>`
- template<typename `_Alloc`>
using `std::__outer_allocator_t` = `decltype(std::declval<_Alloc>().outer_allocator())`

Functions

- template<typename `_Alloc`>
`__outermost_type<_Alloc>::type & std::__outermost` (`_Alloc &a`)
- template<typename `T1`, typename `T2`>
bool `std::operator!=` (const `allocator<T1>` &, const `allocator<T2>` &) noexcept
- template<typename `Tp`>
bool `std::operator!=` (const `allocator<Tp>` &, const `allocator<Tp>` &) noexcept

- `template<typename _OutA1 , typename _OutA2 , typename... _InA>`
`bool std::operator!= (const scoped_allocator_adaptor< _OutA1, _InA... > &__a, const scoped_allocator_↵`
`adaptor< _OutA2, _InA... > &__b) noexcept`
- `template<typename _T1 , typename _T2 >`
`bool std::operator== (const allocator< _T1 > &, const allocator< _T2 > &) noexcept`
- `template<typename _Tp >`
`bool std::operator== (const allocator< _Tp > &, const allocator< _Tp > &) noexcept`
- `template<typename _OutA1 , typename _OutA2 , typename... _InA>`
`bool std::operator== (const scoped_allocator_adaptor< _OutA1, _InA... > &__a, const scoped_allocator_↵`
`adaptor< _OutA2, _InA... > &__b) noexcept`

3.4.1 Detailed Description

Classes encapsulating memory operations.

3.4.2 Typedef Documentation

3.4.2.1 `template<typename _Tp> using std::__allocator_base = typedef __gnu_cxx::new_allocator<_Tp>`

An alias to the base class for `std::allocator`.

Used to set the `std::allocator` base class to `__gnu_cxx::new_allocator`.

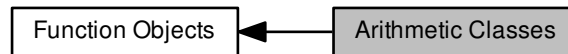
Template Parameters

<code>_Tp</code>	Type of allocated object.
------------------	---------------------------

Definition at line 48 of file `c++allocator.h`.

3.5 Arithmetic Classes

Collaboration diagram for Arithmetic Classes:



Classes

- struct [std::divides< _Tp >](#)
- struct [std::divides< void >](#)
- struct [std::minus< _Tp >](#)
- struct [std::minus< void >](#)
- struct [std::modulus< _Tp >](#)
- struct [std::modulus< void >](#)
- struct [std::multiplies< _Tp >](#)
- struct [std::multiplies< void >](#)
- struct [std::negate< _Tp >](#)
- struct [std::negate< void >](#)
- struct [std::plus< _Tp >](#)

Macros

- `#define __cpp_lib_transparent_operators`

3.5.1 Detailed Description

Because basic math often needs to be done during an algorithm, the library provides functors for those operations. See the documentation for [the base classes](#) for examples of their use.

3.6 Array creation functions

Collaboration diagram for Array creation functions:



Functions

- `template<typename _Tp, size_t _Nm, size_t... _Idx>`
`constexpr array< remove_cv_t< _Tp >, _Nm > std::experimental::fundamentals_v2::__to_array (_Tp(&__↵`
`__a)[_Nm], index_sequence< _Idx... >)`
- `template<typename _Dest = void, typename... _Types>`
`constexpr auto std::experimental::fundamentals_v2::make_array (_Types &&...__t) -> array< conditional_t<`
`is_void_v< _Dest >, common_type_t< _Types... >, _Dest >, sizeof...(_Types)>`
- `template<typename _Tp, size_t _Nm>`
`constexpr array< remove_cv_t< _Tp >, _Nm > std::experimental::fundamentals_v2::to_array (_Tp(&__↵`
`a)[_Nm])`

3.6.1 Detailed Description

Array creation functions as described in N4529, Working Draft, C++ Extensions for Library Fundamentals, Version 2

3.7 Associative

Collaboration diagram for Associative:



Classes

- class `std::map<_Key, _Tp, _Compare, _Alloc>`
- class `std::multimap<_Key, _Tp, _Compare, _Alloc>`
- class `std::multiset<_Key, _Compare, _Alloc>`
- class `std::set<_Key, _Compare, _Alloc>`

3.7.1 Detailed Description

Associative containers allow fast retrieval of data based on keys.

Each container type is parameterized on a `Key` type, and an ordering relation used to sort the elements of the container.

All associative containers must meet certain requirements, summarized in [tables](#).

3.8 Atomics

Classes

- struct `std::__atomic_base<_ITp>`
- struct `std::__atomic_base<_PTp*>`
- struct `std::__atomic_flag_base`
- struct `std::atomic<_Tp>`
- struct `std::atomic<_Tp*>`
- struct `std::atomic<bool>`
- struct `std::atomic<char>`
- struct `std::atomic<char16_t>`
- struct `std::atomic<char32_t>`
- struct `std::atomic<int>`
- struct `std::atomic<long>`
- struct `std::atomic<long long>`
- struct `std::atomic<short>`
- struct `std::atomic<signed char>`
- struct `std::atomic<unsigned char>`
- struct `std::atomic<unsigned int>`
- struct `std::atomic<unsigned long>`
- struct `std::atomic<unsigned long long>`
- struct `std::atomic<unsigned short>`
- struct `std::atomic<wchar_t>`
- struct `std::atomic_flag`

Macros

- `#define ATOMIC_BOOL_LOCK_FREE`
- `#define ATOMIC_CHAR16_T_LOCK_FREE`
- `#define ATOMIC_CHAR32_T_LOCK_FREE`
- `#define ATOMIC_CHAR_LOCK_FREE`
- `#define ATOMIC_FLAG_INIT`
- `#define ATOMIC_INT_LOCK_FREE`
- `#define ATOMIC_LLONG_LOCK_FREE`
- `#define ATOMIC_LONG_LOCK_FREE`
- `#define ATOMIC_POINTER_LOCK_FREE`
- `#define ATOMIC_SHORT_LOCK_FREE`
- `#define ATOMIC_VAR_INIT(_VI)`
- `#define ATOMIC_WCHAR_T_LOCK_FREE`

Typedefs

- typedef unsigned char **std::__atomic_flag_data_type**
- typedef atomic< bool > [std::atomic_bool](#)
- typedef atomic< char > [std::atomic_char](#)
- typedef atomic< char16_t > [std::atomic_char16_t](#)
- typedef atomic< char32_t > [std::atomic_char32_t](#)
- typedef atomic< int > [std::atomic_int](#)
- typedef atomic< int_fast16_t > [std::atomic_int_fast16_t](#)
- typedef atomic< int_fast32_t > [std::atomic_int_fast32_t](#)
- typedef atomic< int_fast64_t > [std::atomic_int_fast64_t](#)
- typedef atomic< int_fast8_t > [std::atomic_int_fast8_t](#)
- typedef atomic< int_least16_t > [std::atomic_int_least16_t](#)
- typedef atomic< int_least32_t > [std::atomic_int_least32_t](#)
- typedef atomic< int_least64_t > [std::atomic_int_least64_t](#)
- typedef atomic< int_least8_t > [std::atomic_int_least8_t](#)
- typedef atomic< intmax_t > [std::atomic_intmax_t](#)
- typedef atomic< intptr_t > [std::atomic_intptr_t](#)
- typedef atomic< long long > [std::atomic_llong](#)
- typedef atomic< long > [std::atomic_long](#)
- typedef atomic< ptrdiff_t > [std::atomic_ptrdiff_t](#)
- typedef atomic< signed char > [std::atomic_schar](#)
- typedef atomic< short > [std::atomic_short](#)
- typedef atomic< size_t > [std::atomic_size_t](#)
- typedef atomic< unsigned char > [std::atomic_uchar](#)
- typedef atomic< unsigned int > [std::atomic_uint](#)
- typedef atomic< uint_fast16_t > [std::atomic_uint_fast16_t](#)
- typedef atomic< uint_fast32_t > [std::atomic_uint_fast32_t](#)
- typedef atomic< uint_fast64_t > [std::atomic_uint_fast64_t](#)
- typedef atomic< uint_fast8_t > [std::atomic_uint_fast8_t](#)
- typedef atomic< uint_least16_t > [std::atomic_uint_least16_t](#)
- typedef atomic< uint_least32_t > [std::atomic_uint_least32_t](#)
- typedef atomic< uint_least64_t > [std::atomic_uint_least64_t](#)
- typedef atomic< uint_least8_t > [std::atomic_uint_least8_t](#)
- typedef atomic< uintmax_t > [std::atomic_uintmax_t](#)
- typedef atomic< uintptr_t > [std::atomic_uintptr_t](#)
- typedef atomic< unsigned long long > [std::atomic_ullong](#)
- typedef atomic< unsigned long > [std::atomic_ulong](#)
- typedef atomic< unsigned short > [std::atomic_ushort](#)
- typedef atomic< wchar_t > [std::atomic_wchar_t](#)
- typedef enum [std::memory_order](#) **std::memory_order**

Enumerations

- enum **__memory_order_modifier** { **__memory_order_mask**, **__memory_order_modifier_mask**, **__memory_order_hle_acquire**, **__memory_order_hle_release** }
- enum [std::memory_order](#) { **memory_order_relaxed**, **memory_order_consume**, **memory_order_acquire**, **memory_order_release**, **memory_order_acq_rel**, **memory_order_seq_cst** }

Functions

- **std::__attribute__** ((__always_inline__)) void atomic_thread_fence(memory_order __m) noexcept
- constexpr memory_order **std::__cmpexch_failure_order** (memory_order __m) noexcept
- constexpr memory_order **std::__cmpexch_failure_order2** (memory_order __m) noexcept
- template<typename _ITp >
bool **std::atomic_compare_exchange_strong** (atomic< _ITp > *__a, _ITp *__i1, _ITp __i2) noexcept
- template<typename _ITp >
bool **std::atomic_compare_exchange_strong** (volatile atomic< _ITp > *__a, _ITp *__i1, _ITp __i2) noexcept
- template<typename _ITp >
bool **std::atomic_compare_exchange_strong_explicit** (atomic< _ITp > *__a, _ITp *__i1, _ITp __i2, memory_order __m1, memory_order __m2) noexcept
- template<typename _ITp >
bool **std::atomic_compare_exchange_strong_explicit** (volatile atomic< _ITp > *__a, _ITp *__i1, _ITp __i2, memory_order __m1, memory_order __m2) noexcept
- template<typename _ITp >
bool **std::atomic_compare_exchange_weak** (atomic< _ITp > *__a, _ITp *__i1, _ITp __i2) noexcept
- template<typename _ITp >
bool **std::atomic_compare_exchange_weak** (volatile atomic< _ITp > *__a, _ITp *__i1, _ITp __i2) noexcept
- template<typename _ITp >
bool **std::atomic_compare_exchange_weak_explicit** (atomic< _ITp > *__a, _ITp *__i1, _ITp __i2, memory_order __m1, memory_order __m2) noexcept
- template<typename _ITp >
bool **std::atomic_compare_exchange_weak_explicit** (volatile atomic< _ITp > *__a, _ITp *__i1, _ITp __i2, memory_order __m1, memory_order __m2) noexcept
- template<typename _ITp >
_ITp **std::atomic_exchange** (atomic< _ITp > *__a, _ITp __i) noexcept
- template<typename _ITp >
_ITp **std::atomic_exchange** (volatile atomic< _ITp > *__a, _ITp __i) noexcept
- template<typename _ITp >
_ITp **std::atomic_exchange_explicit** (atomic< _ITp > *__a, _ITp __i, memory_order __m) noexcept
- template<typename _ITp >
_ITp **std::atomic_exchange_explicit** (volatile atomic< _ITp > *__a, _ITp __i, memory_order __m) noexcept
- template<typename _ITp >
_ITp **std::atomic_fetch_add** (__atomic_base< _ITp > *__a, _ITp __i) noexcept
- template<typename _ITp >
_ITp **std::atomic_fetch_add** (volatile __atomic_base< _ITp > *__a, _ITp __i) noexcept
- template<typename _ITp >
_ITp * **std::atomic_fetch_add** (volatile atomic< _ITp * > *__a, ptrdiff_t __d) noexcept
- template<typename _ITp >
_ITp * **std::atomic_fetch_add** (atomic< _ITp * > *__a, ptrdiff_t __d) noexcept
- template<typename _ITp >
_ITp **std::atomic_fetch_add_explicit** (__atomic_base< _ITp > *__a, _ITp __i, memory_order __m) noexcept
- template<typename _ITp >
_ITp **std::atomic_fetch_add_explicit** (volatile __atomic_base< _ITp > *__a, _ITp __i, memory_order __m) noexcept
- template<typename _ITp >
_ITp * **std::atomic_fetch_add_explicit** (atomic< _ITp * > *__a, ptrdiff_t __d, memory_order __m) noexcept
- template<typename _ITp >
_ITp * **std::atomic_fetch_add_explicit** (volatile atomic< _ITp * > *__a, ptrdiff_t __d, memory_order __m) noexcept
- template<typename _ITp >
_ITp **std::atomic_fetch_and** (__atomic_base< _ITp > *__a, _ITp __i) noexcept

- `template<typename _ITp >`
`_ITp std::atomic_fetch_and (volatile __atomic_base< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_and_explicit (__atomic_base< _ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_and_explicit (volatile __atomic_base< _ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_or (__atomic_base< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_or (volatile __atomic_base< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_or_explicit (__atomic_base< _ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_or_explicit (volatile __atomic_base< _ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_sub (__atomic_base< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_sub (volatile __atomic_base< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp * std::atomic_fetch_sub (volatile atomic< _ITp * > *__a, ptrdiff_t __d) noexcept`
- `template<typename _ITp >`
`_ITp * std::atomic_fetch_sub (atomic< _ITp * > *__a, ptrdiff_t __d) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_sub_explicit (__atomic_base< _ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_sub_explicit (volatile __atomic_base< _ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp * std::atomic_fetch_sub_explicit (volatile atomic< _ITp * > *__a, ptrdiff_t __d, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp * std::atomic_fetch_sub_explicit (atomic< _ITp * > *__a, ptrdiff_t __d, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_xor (__atomic_base< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_xor (volatile __atomic_base< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_xor_explicit (__atomic_base< _ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_xor_explicit (volatile __atomic_base< _ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `void std::atomic_flag_clear (atomic_flag *__a) noexcept`
- `void std::atomic_flag_clear (volatile atomic_flag *__a) noexcept`
- `void std::atomic_flag_clear_explicit (atomic_flag *__a, memory_order __m) noexcept`
- `void std::atomic_flag_clear_explicit (volatile atomic_flag *__a, memory_order __m) noexcept`
- `bool std::atomic_flag_test_and_set (atomic_flag *__a) noexcept`
- `bool std::atomic_flag_test_and_set (volatile atomic_flag *__a) noexcept`
- `bool std::atomic_flag_test_and_set_explicit (atomic_flag *__a, memory_order __m) noexcept`
- `bool std::atomic_flag_test_and_set_explicit (volatile atomic_flag *__a, memory_order __m) noexcept`
- `template<typename _ITp >`
`void std::atomic_init (atomic< _ITp > *__a, _ITp __i) noexcept`

- `template<typename _ITp >`
`void std::atomic_init (volatile atomic< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`bool std::atomic_is_lock_free (const atomic< _ITp > *__a) noexcept`
- `template<typename _ITp >`
`bool std::atomic_is_lock_free (const volatile atomic< _ITp > *__a) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_load (const atomic< _ITp > *__a) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_load (const volatile atomic< _ITp > *__a) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_load_explicit (const atomic< _ITp > *__a, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_load_explicit (const volatile atomic< _ITp > *__a, memory_order __m) noexcept`
- `template<typename _ITp >`
`void std::atomic_store (atomic< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`void std::atomic_store (volatile atomic< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`void std::atomic_store_explicit (atomic< _ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`void std::atomic_store_explicit (volatile atomic< _ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _Tp >`
`_Tp std::kill_dependency (_Tp __y) noexcept`
- `constexpr memory_order std::operator& (memory_order __m, __memory_order_modifier __mod)`
- `constexpr memory_order std::operator| (memory_order __m, __memory_order_modifier __mod)`

3.8.1 Detailed Description

Components for performing atomic operations.

3.8.2 Macro Definition Documentation

3.8.2.1 `#define ATOMIC_BOOL_LOCK_FREE`

Lock-free property.

0 indicates that the types are never lock-free. 1 indicates that the types are sometimes lock-free. 2 indicates that the types are always lock-free.

Definition at line 49 of file `atomic_lockfree_defines.h`.

3.8.3 Typedef Documentation

3.8.3.1 `typedef atomic<bool> std::atomic_bool`

`atomic_bool`

Definition at line 776 of file `atomic`.

3.8.3.2 `typedef atomic<char> std::atomic_char`

`atomic_char`

Definition at line 779 of file `atomic`.

3.8.3.3 `typedef atomic<char16_t> std::atomic_char16_t`

`atomic_char16_t`

Definition at line 815 of file `atomic`.

3.8.3.4 `typedef atomic<char32_t> std::atomic_char32_t`

`atomic_char32_t`

Definition at line 818 of file `atomic`.

3.8.3.5 `typedef atomic<int> std::atomic_int`

`atomic_int`

Definition at line 794 of file `atomic`.

3.8.3.6 `typedef atomic<int_fast16_t> std::atomic_int_fast16_t`

`atomic_int_fast16_t`

Definition at line 853 of file `atomic`.

3.8.3.7 `typedef atomic<int_fast32_t> std::atomic_int_fast32_t`

`atomic_int_fast32_t`

Definition at line 859 of file `atomic`.

3.8.3.8 `typedef atomic<int_fast64_t> std::atomic_int_fast64_t`

`atomic_int_fast64_t`

Definition at line 865 of file `atomic`.

3.8.3.9 `typedef atomic<int_fast8_t> std::atomic_int_fast8_t`

`atomic_int_fast8_t`

Definition at line 847 of file `atomic`.

3.8.3.10 `typedef atomic<int_least16_t> std::atomic_int_least16_t`

`atomic_int_least16_t`

Definition at line 828 of file `atomic`.

3.8.3.11 `typedef atomic<int_least32_t> std::atomic_int_least32_t`

`atomic_int_least32_t`

Definition at line 834 of file `atomic`.

3.8.3.12 `typedef atomic<int_least64_t> std::atomic_int_least64_t`

`atomic_int_least64_t`

Definition at line 840 of file `atomic`.

3.8.3.13 `typedef atomic<int_least8_t> std::atomic_int_least8_t`

`atomic_int_least8_t`

Definition at line 822 of file `atomic`.

3.8.3.14 `typedef atomic<intmax_t> std::atomic_intmax_t`

`atomic_intmax_t`

Definition at line 881 of file `atomic`.

3.8.3.15 `typedef atomic<intptr_t> std::atomic_intptr_t`

`atomic_intptr_t`

Definition at line 872 of file `atomic`.

3.8.3.16 `typedef atomic<long long> std::atomic_llong`

`atomic_llong`

Definition at line 806 of file `atomic`.

3.8.3.17 `typedef atomic<long> std::atomic_long`

`atomic_long`

Definition at line 800 of file `atomic`.

3.8.3.18 `typedef atomic<ptrdiff_t> std::atomic_ptrdiff_t`

`atomic_ptrdiff_t`

Definition at line 887 of file `atomic`.

3.8.3.19 `typedef atomic<signed char> std::atomic_schar`

`atomic_schar`

Definition at line 782 of file `atomic`.

3.8.3.20 `typedef atomic<short> std::atomic_short`

`atomic_short`

Definition at line 788 of file `atomic`.

3.8.3.21 `typedef atomic<size_t> std::atomic_size_t`

`atomic_size_t`

Definition at line 878 of file `atomic`.

3.8.3.22 `typedef atomic<unsigned char> std::atomic_uchar`

`atomic_uchar`

Definition at line 785 of file `atomic`.

3.8.3.23 `typedef atomic<unsigned int> std::atomic_uint`

`atomic_uint`

Definition at line 797 of file `atomic`.

3.8.3.24 `typedef atomic<uint_fast16_t> std::atomic_uint_fast16_t`

`atomic_uint_fast16_t`

Definition at line 856 of file `atomic`.

3.8.3.25 `typedef atomic<uint_fast32_t> std::atomic_uint_fast32_t`

`atomic_uint_fast32_t`

Definition at line 862 of file `atomic`.

3.8.3.26 `typedef atomic<uint_fast64_t> std::atomic_uint_fast64_t``atomic_uint_fast64_t`Definition at line 868 of file `atomic`.**3.8.3.27** `typedef atomic<uint_fast8_t> std::atomic_uint_fast8_t``atomic_uint_fast8_t`Definition at line 850 of file `atomic`.**3.8.3.28** `typedef atomic<uint_least16_t> std::atomic_uint_least16_t``atomic_uint_least16_t`Definition at line 831 of file `atomic`.**3.8.3.29** `typedef atomic<uint_least32_t> std::atomic_uint_least32_t``atomic_uint_least32_t`Definition at line 837 of file `atomic`.**3.8.3.30** `typedef atomic<uint_least64_t> std::atomic_uint_least64_t``atomic_uint_least64_t`Definition at line 843 of file `atomic`.**3.8.3.31** `typedef atomic<uint_least8_t> std::atomic_uint_least8_t``atomic_uint_least8_t`Definition at line 825 of file `atomic`.**3.8.3.32** `typedef atomic<uintmax_t> std::atomic_uintmax_t``atomic_uintmax_t`Definition at line 884 of file `atomic`.**3.8.3.33** `typedef atomic<uintptr_t> std::atomic_uintptr_t``atomic_uintptr_t`Definition at line 875 of file `atomic`.

3.8.3.34 `typedef atomic<unsigned long long> std::atomic_ullong`

`atomic_ullong`

Definition at line 809 of file `atomic`.

3.8.3.35 `typedef atomic<unsigned long> std::atomic_ulong`

`atomic_ulong`

Definition at line 803 of file `atomic`.

3.8.3.36 `typedef atomic<unsigned short> std::atomic_ushort`

`atomic_ushort`

Definition at line 791 of file `atomic`.

3.8.3.37 `typedef atomic<wchar_t> std::atomic_wchar_t`

`atomic_wchar_t`

Definition at line 812 of file `atomic`.

3.8.3.38 `typedef enum std::memory_order std::memory_order`

Enumeration for `memory_order`.

3.8.4 Enumeration Type Documentation**3.8.4.1** `enum std::memory_order`

Enumeration for `memory_order`.

Definition at line 55 of file `atomic_base.h`.

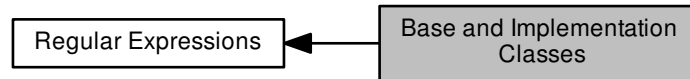
3.8.5 Function Documentation**3.8.5.1** `template<typename _Tp> _Tp std::kill_dependency (_Tp__y) [inline], [noexcept]`

`kill_dependency`

Definition at line 111 of file `atomic_base.h`.

3.9 Base and Implementation Classes

Collaboration diagram for Base and Implementation Classes:



Classes

- struct `std::__detail::BracketMatcher<_TraitsT, __icase, __collate >`
- class `std::__detail::Compiler<_TraitsT >`
- class `std::__detail::Executor<_Bilter, _Alloc, _TraitsT, __dfs_mode >`
- class `std::__detail::Scanner<_CharT >`
- class `std::__detail::StateSeq<_TraitsT >`

Typedefs

- template<typename _Iter, typename _TraitsT >
using `std::__detail::__disable_if_contiguous_normal_iter` = typename enable_if< !__is_contiguous_↵
normal_iter<_Iter >::value, `std::shared_ptr`< const _NFA<_TraitsT >> >::type
- template<typename _Iter, typename _TraitsT >
using `std::__detail::__enable_if_contiguous_normal_iter` = typename enable_if< __is_contiguous_normal_↵
_iter<_Iter >::value, `std::shared_ptr`< const _NFA<_TraitsT >> >::type
- template<typename _CharT >
using `std::__detail::Matcher` = std::function< bool(_CharT)>
- typedef long `std::__detail::StatIdT`

Enumerations

- enum `std::__detail::Opcode` : int {
_S_opcode_unknown, _S_opcode_alternative, _S_opcode_repeat, _S_opcode_backref,
_S_opcode_line_begin_assertion, _S_opcode_line_end_assertion, _S_opcode_word_boundary, _S_↵
opcode_subexpr_lookahead,
_S_opcode_subexpr_begin, _S_opcode_subexpr_end, _S_opcode_dummy, _S_opcode_match,
_S_opcode_accept }

Functions

- template<typename _FwdIter, typename _TraitsT >
__enable_if_contiguous_normal_iter<_FwdIter, _TraitsT > `std::__detail::__compile_nfa` (_FwdIter __first, _↵
FwdIter __last, const typename _TraitsT::locale_type &__loc, regex_constants::syntax_option_type __flags)

Variables

- static const _StateIdT **std::__detail::_S_invalid_state_id**

3.9.1 Detailed Description

3.9.2 Enumeration Type Documentation

3.9.2.1 enum std::__detail::_Opcode : int

Operation codes that define the type of transitions within the base NFA that represents the regular expression.

Definition at line 56 of file regex_automaton.h.

3.10 Base and Implementation Classes

Collaboration diagram for Base and Implementation Classes:



Classes

- struct `std::__detail::__Default_ranged_hash`
- struct `std::__detail::__Equal_helper< _Key, _Value, _ExtractKey, _Equal, _HashCodeType, __cache_hash_code >`
- struct `std::__detail::__Equal_helper< _Key, _Value, _ExtractKey, _Equal, _HashCodeType, false >`
- struct `std::__detail::__Equal_helper< _Key, _Value, _ExtractKey, _Equal, _HashCodeType, true >`
- struct `std::__detail::__Equality< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, _Unique_keys >`
- struct `std::__detail::__Equality< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, false >`
- struct `std::__detail::__Equality< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, true >`
- struct `std::__detail::__Equality_base`
- struct `std::__detail::__Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache_hash_code >`
- struct `std::__detail::__Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Default_ranged_hash, false >`
- struct `std::__detail::__Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Default_ranged_hash, true >`
- struct `std::__detail::__Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, false >`
- struct `std::__detail::__Hash_node< _Value, _Cache_hash_code >`
- struct `std::__detail::__Hash_node< _Value, false >`
- struct `std::__detail::__Hash_node< _Value, true >`
- struct `std::__detail::__Hash_node_base`
- struct `std::__detail::__Hash_node_value_base< _Value >`
- struct `std::__detail::__Hashtable_alloc< _NodeAlloc >`
- struct `std::__detail::__Hashtable_base< _Key, _Value, _ExtractKey, _Equal, _H1, _H2, _Hash, _Traits >`
- struct `std::__detail::__Hashtable_ebo_helper< _Nm, _Tp, __use_ebo >`
- struct `std::__detail::__Hashtable_ebo_helper< _Nm, _Tp, false >`
- struct `std::__detail::__Hashtable_ebo_helper< _Nm, _Tp, true >`
- struct `std::__detail::__Hashtable_traits< _Cache_hash_code, _Constant_iterators, _Unique_keys >`
- struct `std::__detail::__Insert< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, __ Traits, _Constant_iterators, _Unique_keys >`
- struct `std::__detail::__Insert< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, __ Traits, false, _Unique_keys >`
- struct `std::__detail::__Insert< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, __ Traits, true, false >`

- `struct std::__detail::Insert< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, true, true >`
- `struct std::__detail::Insert_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits >`
- `struct std::__detail::Local_const_iterator< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __constant_iterators, __cache >`
- `struct std::__detail::Local_iterator< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __constant_iterators, __cache >`
- `struct std::__detail::Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache_hash_code >`
- `struct std::__detail::Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, true >`
- `struct std::__detail::Map_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, _Unique_keys >`
- `struct std::__detail::Map_base< _Key, _Pair, _Alloc, _Select1st, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, false >`
- `struct std::__detail::Map_base< _Key, _Pair, _Alloc, _Select1st, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, true >`
- `struct std::__detail::Mod_range_hashing`
- `struct std::__detail::Node_const_iterator< _Value, __constant_iterators, __cache >`
- `struct std::__detail::Node_iterator< _Value, __constant_iterators, __cache >`
- `struct std::__detail::Node_iterator_base< _Value, _Cache_hash_code >`
- `struct std::__detail::Prime_rehash_policy`
- `struct std::__detail::Rehash_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits >`
- `struct std::__detail::Rehash_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _Prime_rehash_policy, _Traits >`
- `class std::__detail::Hashtable< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits >`

Typedefs

- `template<typename _Key, typename _Value, typename _ExtractKey, typename _H1, typename _H2, typename _Hash >
using std::__detail::__hash_code_for_local_iter = _Hash_code_storage< _Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, false >>`

Functions

- `template<class _Iterator >
std::iterator_traits< _Iterator >::difference_type std::__detail::__distance_fw (_Iterator __first, _Iterator __last, std::input_iterator_tag)`
- `template<class _Iterator >
std::iterator_traits< _Iterator >::difference_type std::__detail::__distance_fw (_Iterator __first, _Iterator __last, std::forward_iterator_tag)`
- `template<class _Iterator >
std::iterator_traits< _Iterator >::difference_type std::__detail::__distance_fw (_Iterator __first, _Iterator __last)`
- `__bucket_type * std::__detail::__Hashtable_alloc< _NodeAlloc >::M_allocate_buckets (std::size_t __n)`
- `template<typename... _Args>
__node_type * std::__detail::__Hashtable_alloc< _NodeAlloc >::M_allocate_node (_Args &&... __args)`
- `void std::__detail::__Hashtable_alloc< _NodeAlloc >::M_deallocate_buckets (__bucket_type *, std::size_t __n)`
- `void std::__detail::__Hashtable_alloc< _NodeAlloc >::M_deallocate_node (__node_type * __n)`
- `void std::__detail::__Hashtable_alloc< _NodeAlloc >::M_deallocate_nodes (__node_type * __n)`

- `template<typename _InputIterator, typename _NodeGetter >`
`void std::__detail::Insert_base<_Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _Rehash, _Policy, _Traits >::M_insert_range` (_InputIterator __first, _InputIterator __last, const _NodeGetter &)
- `template<typename _Uiterator >`
`static bool std::__detail::Equality_base::S_is_permutation` (_Uiterator, _Uiterator, _Uiterator)
- `template<typename _Value, bool _Cache_hash_code>`
`bool std::__detail::operator!=` (const _Node_iterator_base< _Value, _Cache_hash_code > &__x, const _Node_iterator_base< _Value, _Cache_hash_code > &__y) noexcept
- `template<typename _Key, typename _Value, typename _ExtractKey, typename _H1, typename _H2, typename _Hash, bool __cache>`
`bool std::__detail::operator!=` (const _Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache > &__x, const _Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache > &__y)
- `template<typename _Value, bool _Cache_hash_code>`
`bool std::__detail::operator==` (const _Node_iterator_base< _Value, _Cache_hash_code > &__x, const _Node_iterator_base< _Value, _Cache_hash_code > &__y) noexcept
- `template<typename _Key, typename _Value, typename _ExtractKey, typename _H1, typename _H2, typename _Hash, bool __cache>`
`bool std::__detail::operator==` (const _Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache > &__x, const _Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache > &__y)

3.10.1 Detailed Description

3.11 Base and Policy Classes

Collaboration diagram for Base and Policy Classes:



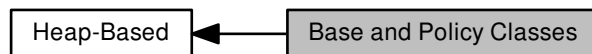
Classes

- [class `__gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >`](#)
- [class `__gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >`](#)
- [class `__gnu_pbds::detail::rb_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >`](#)
- [class `__gnu_pbds::detail::splay_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >`](#)

3.11.1 Detailed Description

3.12 Base and Policy Classes

Collaboration diagram for Base and Policy Classes:



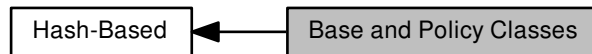
Classes

- class `__gnu_pbds::detail::binary_heap< Value_Type, Cmp_Fn, _Alloc >`
- class `__gnu_pbds::detail::binomial_heap< Value_Type, Cmp_Fn, _Alloc >`
- class `__gnu_pbds::detail::pairing_heap< Value_Type, Cmp_Fn, _Alloc >`
- class `__gnu_pbds::detail::rc_binomial_heap< Value_Type, Cmp_Fn, _Alloc >`
- class `__gnu_pbds::detail::thin_heap< Value_Type, Cmp_Fn, _Alloc >`

3.12.1 Detailed Description

3.13 Base and Policy Classes

Collaboration diagram for Base and Policy Classes:



Classes

- [class `__gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy >`](#)
- [class `__gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >`](#)

3.13.1 Detailed Description

3.14 Bernoulli Distributions

Collaboration diagram for Bernoulli Distributions:



Classes

- class `std::bernoulli_distribution`
- class `std::binomial_distribution< _IntType >`
- class `std::geometric_distribution< _IntType >`
- class `std::negative_binomial_distribution< _IntType >`

Functions

- `bool std::operator!= (const std::bernoulli_distribution &__d1, const std::bernoulli_distribution &__d2)`
- `template<typename _IntType >
bool std::operator!= (const std::binomial_distribution< _IntType > &__d1, const std::binomial_distribution< _IntType > &__d2)`
- `template<typename _IntType >
bool std::operator!= (const std::geometric_distribution< _IntType > &__d1, const std::geometric_distribution< _IntType > &__d2)`
- `template<typename _IntType >
bool std::operator!= (const std::negative_binomial_distribution< _IntType > &__d1, const std::negative_binomial_distribution< _IntType > &__d2)`
- `template<typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const std::bernoulli_distribution &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const std::geometric_distribution< _IntType > &__x)`
- `template<typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is, std::bernoulli_distribution &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is, std::geometric_distribution< _IntType > &__x)`

3.14.1 Detailed Description

3.14.2 Function Documentation

3.14.2.1 `bool std::operator!= (const std::bernoulli_distribution & __d1, const std::bernoulli_distribution & __d2)`
`[inline]`

Return true if two Bernoulli distributions have different parameters.

Definition at line 3564 of file random.h.

3.14.2.2 `template<typename _IntType> bool std::operator!= (const std::binomial_distribution<_IntType> & __d1, const std::binomial_distribution<_IntType> & __d2)` `[inline]`

Return true if two binomial distributions are different.

Definition at line 3830 of file random.h.

3.14.2.3 `template<typename _IntType> bool std::operator!= (const std::geometric_distribution<_IntType> & __d1, const std::geometric_distribution<_IntType> & __d2)` `[inline]`

Return true if two geometric distributions have different parameters.

Definition at line 3999 of file random.h.

References `std::__detail::operator>>()`.

3.14.2.4 `template<typename _IntType> bool std::operator!= (const std::negative_binomial_distribution<_IntType> & __d1, const std::negative_binomial_distribution<_IntType> & __d2)` `[inline]`

Return true if two negative binomial distributions are different.

Definition at line 4244 of file random.h.

3.14.2.5 `template<typename _CharT, typename _Traits> std::basic_ostream<_CharT, _Traits> & std::operator<< (std::basic_ostream<_CharT, _Traits> & __os, const std::bernoulli_distribution & __x)`

Inserts a `bernoulli_distribution` random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>bernoulli_distribution</code> random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

Definition at line 998 of file bits/random.tcc.

References `std::numeric_limits<_Tp>::epsilon()`, `std::ios_base::flags()`, `std::left()`, `std::log()`, `std::numeric_limits<_Tp>::max()`, `std::geometric_distribution<_IntType>::operator()()`, and `std::scientific()`.

3.14.2.6 `template<typename _IntType, typename _CharT, typename _Traits> std::basic_ostream<_CharT, _Traits> & std::operator<< (std::basic_ostream<_CharT, _Traits> & __os, const std::geometric_distribution<_IntType> & __x)`

Inserts a `geometric_distribution` random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>geometric_distribution</code> random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

Definition at line 1080 of file bits/random.tcc.

References `std::ios_base::flags()`, `std::left()`, and `std::scientific()`.

3.14.2.7 `template<typename _CharT, typename _Traits> std::basic_istream<_CharT, _Traits> & std::operator>> (std::basic_istream<_CharT, _Traits> & __is, std::bernoulli_distribution & __x)`

Extracts a `bernoulli_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>bernoulli_distribution</code> random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

Definition at line 3594 of file random.h.

References `std::bernoulli_distribution::param()`.

3.14.2.8 `template<typename _IntType, typename _CharT, typename _Traits> std::basic_istream<_CharT, _Traits> & std::operator>> (std::basic_istream<_CharT, _Traits> & __is, std::geometric_distribution<_IntType> & __x)`

Extracts a `geometric_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>_↔ _is</code>	An input stream.
<code>_↔ _x</code>	A <code>geometric_distribution</code> random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

Definition at line 1104 of file `bits/random.tcc`.

References `std::ios_base::flags()`, `std::negative_binomial_distribution< _IntType >::operator()()`, `std::geometric_↔
distribution< _IntType >::param()`, and `std::skipws()`.

3.15 Binary Search

Collaboration diagram for Binary Search:



Functions

- `template<typename _ForwardIterator, typename _Tp >`
`bool std::binary_search (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`bool std::binary_search (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Tp >`
`pair< _ForwardIterator, _ForwardIterator > std::equal_range (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`pair< _ForwardIterator, _ForwardIterator > std::equal_range (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Tp >`
`_ForwardIterator std::lower_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`_ForwardIterator std::lower_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Tp >`
`_ForwardIterator std::upper_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`_ForwardIterator std::upper_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)`

3.15.1 Detailed Description

These algorithms are variations of a classic binary search, and all assume that the sequence being searched is already sorted.

The number of comparisons will be logarithmic (and as few as possible). The number of steps through the sequence will be logarithmic for random-access iterators (e.g., pointers), and linear otherwise.

The LWG has passed Defect Report 270, which notes: *The proposed resolution reinterprets binary search. Instead of thinking about searching for a value in a sorted range, we view that as an important special case of a more general algorithm: searching for the partition point in a partitioned range. We also add a guarantee that the old wording did not: we ensure that the upper bound is no earlier than the lower bound, that the pair returned by equal_range is a valid range, and that the first part of that pair is the lower bound.*

The actual effect of the first sentence is that a comparison functor passed by the user doesn't necessarily need to induce a strict weak ordering relation. Rather, it partitions the range.

3.15.2 Function Documentation

3.15.2.1 `template<typename _ForwardIterator, typename _Tp> bool std::binary_search (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __val)`

Determines whether an element exists in a range.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__val</code>	The search term.

Returns

True if `__val` (or its equivalent) is in `[__first, __last]`.

Note that this does not actually return an iterator to `__val`. For that, use `std::find` or a container's specialized find member functions.

Definition at line 2244 of file `stl_algo.h`.

3.15.2.2 `template<typename _ForwardIterator, typename _Tp, typename _Compare> bool std::binary_search (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __val, _Compare __comp)`

Determines whether an element exists in a range.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__val</code>	The search term.
<code>__comp</code>	A functor to use for comparisons.

Returns

True if `__val` (or its equivalent) is in `[__first, __last]`.

Note that this does not actually return an iterator to `__val`. For that, use `std::find` or a container's specialized find member functions.

The comparison function should have the same effects on ordering as the function used for the initial sort.

Definition at line 2277 of file `stl_algo.h`.

3.15.2.3 `template<typename _ForwardIterator, typename _Tp> pair<_ForwardIterator, _ForwardIterator> std::equal_range (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __val) [inline]`

Finds the largest subrange in which `__val` could be inserted at any place in it without changing the ordering.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__val</code>	The search term.

Returns

An pair of iterators defining the subrange.

This is equivalent to

```
std::make_pair(lower_bound(__first, __last, __val),
               upper_bound(__first, __last, __val))
```

but does not actually call those functions.

Definition at line 2175 of file `stl_algo.h`.

```
3.15.2.4 template<typename _ForwardIterator, typename _Tp, typename _Compare > pair<_ForwardIterator, _ForwardIterator>
std::equal_range ( _ForwardIterator __first, _ForwardIterator __last, const _Tp & __val, _Compare __comp )
[inline]
```

Finds the largest subrange in which `__val` could be inserted at any place in it without changing the ordering.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__val</code>	The search term.
<code>__comp</code>	A functor to use for comparisons.

Returns

An pair of iterators defining the subrange.

This is equivalent to

```
std::make_pair(lower_bound(__first, __last, __val, __comp),
               upper_bound(__first, __last, __val, __comp))
```

but does not actually call those functions.

Definition at line 2211 of file `stl_algo.h`.

```
3.15.2.5 template<typename _ForwardIterator, typename _Tp > _ForwardIterator std::lower_bound ( _ForwardIterator __first,
_FForwardIterator __last, const _Tp & __val ) [inline]
```

Finds the first position in which `val` could be inserted without changing the ordering.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__val</code>	The search term.

Returns

An iterator pointing to the first element *not less than* `val`, or `end()` if every element is less than `val`.

Definition at line 984 of file `stl_algobase.h`.

Referenced by `std::operator>>()`.

```
3.15.2.6 template<typename _ForwardIterator, typename _Tp, typename _Compare> _ForwardIterator std::lower_bound (
    _ForwardIterator __first, _ForwardIterator __last, const _Tp & __val, _Compare __comp ) [inline]
```

Finds the first position in which `__val` could be inserted without changing the ordering.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__val</code>	The search term.
<code>__comp</code>	A functor to use for comparisons.

Returns

An iterator pointing to the first element *not less than* `__val`, or `end()` if every element is less than `__val`.

The comparison function should have the same effects on ordering as the function used for the initial sort.

Definition at line 2020 of file `stl_algo.h`.

References `std::advance()`, and `std::distance()`.

```
3.15.2.7 template<typename _ForwardIterator, typename _Tp> _ForwardIterator std::upper_bound ( _ForwardIterator __first,
    _ForwardIterator __last, const _Tp & __val ) [inline]
```

Finds the last position in which `__val` could be inserted without changing the ordering.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__val</code>	The search term.

Returns

An iterator pointing to the first element greater than `__val`, or `end()` if no elements are greater than `__val`.

Definition at line 2074 of file `stl_algo.h`.

3.15.2.8 `template<typename _ForwardIterator, typename _Tp, typename _Compare> _ForwardIterator std::upper_bound (`
`_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp) [inline]`

Finds the last position in which `__val` could be inserted without changing the ordering.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__val</code>	The search term.
<code>__comp</code>	A functor to use for comparisons.

Returns

An iterator pointing to the first element greater than `__val`, or `end()` if no elements are greater than `__val`.

The comparison function should have the same effects on ordering as the function used for the initial sort.

Definition at line 2104 of file `stl_algo.h`.

References `std::advance()`, and `std::distance()`.

3.16 Binder Classes

Collaboration diagram for Binder Classes:



Namespaces

- [std::placeholders](#)

Classes

- struct [std::_Placeholder<_Num>](#)
- class [std::binder1st<_Operation>](#)
- class [std::binder2nd<_Operation>](#)
- struct [std::is_bind_expression<_Tp>](#)
- struct [std::is_bind_expression<_Bind<_Signature>>](#)
- struct [std::is_bind_expression<_Bind_result<_Result, _Signature>>](#)
- struct [std::is_bind_expression<const _Bind<_Signature>>](#)
- struct [std::is_bind_expression<const _Bind_result<_Result, _Signature>>](#)
- struct [std::is_bind_expression<const volatile _Bind<_Signature>>](#)
- struct [std::is_bind_expression<const volatile _Bind_result<_Result, _Signature>>](#)
- struct [std::is_bind_expression<volatile _Bind<_Signature>>](#)
- struct [std::is_bind_expression<volatile _Bind_result<_Result, _Signature>>](#)
- struct [std::is_placeholder<_Tp>](#)
- struct [std::is_placeholder<_Placeholder<_Num>>](#)

Functions

- template<typename _Func, typename... _BoundArgs>
[_Bind_helper<__is_socketlike<_Func>::value, _Func, _BoundArgs...>::type](#) [std::bind](#) (_Func &&__f, _↵
 BoundArgs &&...__args)
- template<typename _Result, typename _Func, typename... _BoundArgs>
[_Bindres_helper<_Result, _Func, _BoundArgs...>::type](#) [std::bind](#) (_Func &&__f, _BoundArgs &&...__args)
- template<typename _Operation, typename _Tp>
[binder1st<_Operation>](#) [std::bind1st](#) (const _Operation &__fn, const _Tp &__x)
- template<typename _Operation, typename _Tp>
[binder2nd<_Operation>](#) [std::bind2nd](#) (const _Operation &__fn, const _Tp &__x)

3.16.1 Detailed Description

Binders turn functions/functors with two arguments into functors with a single argument, storing an argument to be applied later. For example, a variable `B` of type `binder1st` is constructed from a functor `f` and an argument `x`. Later, `B`'s `operator()` is called with a single argument `y`. The return value is the value of `f(x, y)`. `B` can be *called* with various arguments (`y1, y2, ...`) and will in turn call `f(x, y1), f(x, y2), ...`

The function `bind1st` is provided to save some typing. It takes the function and an argument as parameters, and returns an instance of `binder1st`.

The type `binder2nd` and its creator function `bind2nd` do the same thing, but the stored argument is passed as the second parameter instead of the first, e.g., `bind2nd(std::minus<float>(), 1.3)` will create a functor whose `operator()` accepts a floating-point number, subtracts 1.3 from it, and returns the result. (If `bind1st` had been used, the functor would perform `1.3 - x` instead.

Creator-wrapper functions like `bind1st` are intended to be used in calling algorithms. Their return values will be temporary objects. (The goal is to not require you to type names like `std::binder1st<std::plus<int>>` for declaring a variable to hold the return value from `bind1st(std::plus<int>(), 5)`).

These become more useful when combined with the composition functions.

These functions are deprecated in C++11 and can be replaced by `std::bind` (or `std::tr1::bind`) which is more powerful and flexible, supporting functions with any number of arguments. Uses of `bind1st` can be replaced by `std::bind(f, x, std::placeholders::_1)` and `bind2nd` by `std::bind(f, std::placeholders::_1, x)`.

3.16.2 Function Documentation

3.16.2.1 `template<typename _Func, typename... _BoundArgs> _Bind_helper<__is_socketlike<_Func>::value, _Func, _BoundArgs...>::type std::bind(_Func && __f, _BoundArgs &&... __args) [inline]`

Function template for `std::bind`.

Definition at line 1322 of file `functional`.

3.16.2.2 `template<typename _Result, typename _Func, typename... _BoundArgs> _Bindres_helper<_Result, _Func, _BoundArgs...>::type std::bind(_Func && __f, _BoundArgs &&... __args) [inline]`

Function template for `std::bind<R>`.

Definition at line 1350 of file `functional`.

3.16.2.3 `template<typename _Operation, typename _Tp> binder1st<_Operation> std::bind1st(const _Operation & __fn, const _Tp & __x) [inline]`

One of the [binder functors](#).

Definition at line 135 of file `binders.h`.

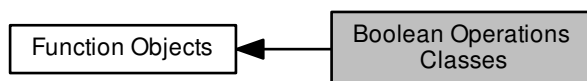
3.16.2.4 `template<typename _Operation, typename _Tp> binder2nd<_Operation> std::bind2nd(const _Operation & __fn, const _Tp & __x) [inline]`

One of the [binder functors](#).

Definition at line 170 of file `binders.h`.

3.17 Boolean Operations Classes

Collaboration diagram for Boolean Operations Classes:



Classes

- struct `std::logical_and< _Tp >`
- struct `std::logical_and< void >`
- struct `std::logical_not< _Tp >`
- struct `std::logical_not< void >`
- struct `std::logical_or< _Tp >`
- struct `std::logical_or< void >`

3.17.1 Detailed Description

Here are wrapper functors for Boolean operations: `&&`, `||`, and `!`.

3.18 Branch-Based

Collaboration diagram for Branch-Based:



Modules

- [Base and Policy Classes](#)

Classes

- [class `__gnu_pbds::basic_branch< Key, Mapped, Tag, Node_Update, Policy_Tl, _Alloc >`](#)
- [class `__gnu_pbds::tree< Key, Mapped, Cmp_Fn, Tag, Node_Update, _Alloc >`](#)
- [class `__gnu_pbds::trie< Key, Mapped, _ATraits, Tag, Node_Update, _Alloc >`](#)

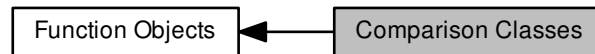
Macros

- `#define PB_DS_BRANCH_BASE`
- `#define PB_DS_TREE_BASE`
- `#define PB_DS_TREE_NODE_AND_IT_TRAITS`
- `#define PB_DS_TRIE_BASE`
- `#define PB_DS_TRIE_NODE_AND_IT_TRAITS`

3.18.1 Detailed Description

3.19 Comparison Classes

Collaboration diagram for Comparison Classes:



Classes

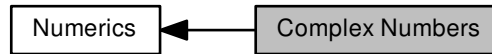
- struct `std::equal_to<_Tp>`
- struct `std::equal_to<void>`
- struct `std::greater<_Tp>`
- struct `std::greater<void>`
- struct `std::greater_equal<_Tp>`
- struct `std::greater_equal<void>`
- struct `std::less<_Tp>`
- struct `std::less<void>`
- struct `std::less_equal<_Tp>`
- struct `std::less_equal<void>`
- struct `std::not_equal_to<_Tp>`
- struct `std::not_equal_to<void>`

3.19.1 Detailed Description

The library provides six wrapper functors for all the basic comparisons in C++, like `<`.

3.20 Complex Numbers

Collaboration diagram for Complex Numbers:



Classes

- struct `std::complex< _Tp >`
- struct `std::complex< double >`
- struct `std::complex< float >`
- struct `std::complex< long double >`

Functions

- constexpr `std::complex< float >::complex` (const complex< double > &)
- constexpr `std::complex< float >::complex` (const complex< long double > &)
- constexpr `std::complex< double >::complex` (const complex< long double > &)
- template<typename _Tp >
_Tp `std::__complex_abs` (const complex< _Tp > &__z)
- template<typename _Tp >
_Tp `std::__complex_arg` (const complex< _Tp > &__z)
- template<typename _Tp >
complex< _Tp > `std::__complex_cos` (const complex< _Tp > &__z)
- template<typename _Tp >
complex< _Tp > `std::__complex_cosh` (const complex< _Tp > &__z)
- template<typename _Tp >
complex< _Tp > `std::__complex_exp` (const complex< _Tp > &__z)
- template<typename _Tp >
complex< _Tp > `std::__complex_log` (const complex< _Tp > &__z)
- template<typename _Tp >
complex< _Tp > `std::__complex_pow` (const complex< _Tp > &__x, const complex< _Tp > &__y)
- template<typename _Tp >
complex< _Tp > `std::__complex_pow_unsigned` (complex< _Tp > __x, unsigned __n)
- template<typename _Tp >
complex< _Tp > `std::__complex_sin` (const complex< _Tp > &__z)
- template<typename _Tp >
complex< _Tp > `std::__complex_sinh` (const complex< _Tp > &__z)
- template<typename _Tp >
complex< _Tp > `std::__complex_sqrt` (const complex< _Tp > &__z)
- template<typename _Tp >
complex< _Tp > `std::__complex_tan` (const complex< _Tp > &__z)

- `template<typename _Tp >`
`complex< _Tp > std::__complex_tanh (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`_Tp std::abs (const complex< _Tp > &)`
- `template<typename _Tp >`
`_Tp std::arg (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::conj (const complex< _Tp > &)`
- `template<typename _Tp >`
`std::complex< _Tp > std::tr1::conj (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< typename __gnu_cxx::__promote< _Tp >::__type > std::tr1::conj (_Tp __x)`
- `template<typename _Tp >`
`complex< _Tp > std::cos (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::cosh (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::exp (const complex< _Tp > &)`
- `template<typename _Tp >`
`std::complex< _Tp > std::tr1::fabs (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`constexpr _Tp std::imag (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > std::log (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::log10 (const complex< _Tp > &)`
- `template<typename _Tp >`
`_Tp std::norm (const complex< _Tp > &)`
- `complex< _Tp > & std::complex< _Tp >::operator*= (const _Tp &)`
- `template<typename _Up >`
`complex< _Tp > & std::complex< _Tp >::operator*= (const complex< _Up > &)`
- `template<typename _Tp >`
`complex< _Tp > std::operator+ (const complex< _Tp > &__x)`
- `template<typename _Up >`
`complex< _Tp > & std::complex< _Tp >::operator+= (const complex< _Up > &)`
- `template<typename _Tp >`
`complex< _Tp > std::operator- (const complex< _Tp > &__x)`
- `template<typename _Up >`
`complex< _Tp > & std::complex< _Tp >::operator-= (const complex< _Up > &)`
- `complex< _Tp > & std::complex< _Tp >::operator/= (const _Tp &)`
- `template<typename _Up >`
`complex< _Tp > & std::complex< _Tp >::operator/= (const complex< _Up > &)`
- `template<typename _Tp, typename _CharT, class _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, const complex< _Tp > &__x)`
- `complex< _Tp > & std::complex< _Tp >::operator= (const _Tp &)`
- `template<typename _Up >`
`complex< _Tp > & std::complex< _Tp >::operator= (const complex< _Up > &)`
- `template<typename _Tp, typename _CharT, class _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__is, complex< _Tp > &__x)`
- `template<typename _Tp >`
`complex< _Tp > std::polar (const _Tp &, const _Tp &=0)`

- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::tr1::polar` (const _Tp &__rho,
const _Up &__theta)
- `template<typename _Tp >`
`complex< _Tp > std::pow` (const complex< _Tp > &, int)
- `template<typename _Tp >`
`complex< _Tp > std::pow` (const complex< _Tp > &, const _Tp &)
- `template<typename _Tp >`
`complex< _Tp > std::pow` (const complex< _Tp > &, const complex< _Tp > &)
- `template<typename _Tp >`
`complex< _Tp > std::pow` (const _Tp &, const complex< _Tp > &)
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::tr1::pow` (const std::complex<
_Tp > &__x, const _Up &__y)
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::tr1::pow` (const _Tp &__x,
const std::complex< _Up > &__y)
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::tr1::pow` (const std::complex<
_Tp > &__x, const std::complex< _Up > &__y)
- `template<typename _Tp >`
`std::complex< _Tp > std::tr1::pow` (const std::complex< _Tp > &__x, const _Tp &__y)
- `template<typename _Tp >`
`std::complex< _Tp > std::tr1::pow` (const _Tp &__x, const std::complex< _Tp > &__y)
- `template<typename _Tp >`
`std::complex< _Tp > std::tr1::pow` (const std::complex< _Tp > &__x, const std::complex< _Tp > &__y)
- `template<typename _Tp >`
`constexpr _Tp std::real` (const complex< _Tp > &__z)
- `template<typename _Tp >`
`complex< _Tp > std::sin` (const complex< _Tp > &)
- `template<typename _Tp >`
`complex< _Tp > std::sinh` (const complex< _Tp > &)
- `template<typename _Tp >`
`complex< _Tp > std::sqrt` (const complex< _Tp > &)
- `template<typename _Tp >`
`complex< _Tp > std::tan` (const complex< _Tp > &)
- `template<typename _Tp >`
`complex< _Tp > std::tanh` (const complex< _Tp > &)

- `template<typename _Tp >`
`complex< _Tp > std::operator+` (const complex< _Tp > &__x, const complex< _Tp > &__y)
- `template<typename _Tp >`
`complex< _Tp > std::operator+` (const complex< _Tp > &__x, const _Tp &__y)
- `template<typename _Tp >`
`complex< _Tp > std::operator+` (const _Tp &__x, const complex< _Tp > &__y)

- `template<typename _Tp >`
`complex< _Tp > std::operator-` (const complex< _Tp > &__x, const complex< _Tp > &__y)
- `template<typename _Tp >`
`complex< _Tp > std::operator-` (const complex< _Tp > &__x, const _Tp &__y)
- `template<typename _Tp >`
`complex< _Tp > std::operator-` (const _Tp &__x, const complex< _Tp > &__y)

- `template<typename _Tp >`
`complex< _Tp > std::operator* (const complex< _Tp > &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`complex< _Tp > std::operator* (const complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp >`
`complex< _Tp > std::operator* (const _Tp &__x, const complex< _Tp > &__y)`

- `template<typename _Tp >`
`complex< _Tp > std::operator/ (const complex< _Tp > &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`complex< _Tp > std::operator/ (const complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp >`
`complex< _Tp > std::operator/ (const _Tp &__x, const complex< _Tp > &__y)`

- `template<typename _Tp >`
`constexpr bool std::operator== (const complex< _Tp > &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`constexpr bool std::operator== (const complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp >`
`constexpr bool std::operator== (const _Tp &__x, const complex< _Tp > &__y)`

- `template<typename _Tp >`
`constexpr bool std::operator!= (const complex< _Tp > &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`constexpr bool std::operator!= (const complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp >`
`constexpr bool std::operator!= (const _Tp &__x, const complex< _Tp > &__y)`

3.20.1 Detailed Description

Classes and functions for complex numbers.

3.20.2 Function Documentation

3.20.2.1 `template<typename _Tp > _Tp std::abs (const complex< _Tp > &__z) [inline]`

Return magnitude of `z`.

Definition at line 601 of file `complex`.

Referenced by `std::binomial_distribution< _IntType >::operator()()`, and `std::poisson_distribution< _IntType >::operator()()`.

3.20.2.2 `template<typename _Tp > _Tp std::arg (const complex< _Tp > &__z) [inline]`

Return phase angle of `z`.

Definition at line 628 of file `complex`.

3.20.2.3 `template<typename _Tp> complex<_Tp> std::conj (const complex<_Tp> &__z) [inline]`

Return complex conjugate of z .

Definition at line 677 of file `complex`.

3.20.2.4 `template<typename _Tp> complex<_Tp> std::cos (const complex<_Tp> &__z) [inline]`

Return complex cosine of z .

Definition at line 709 of file `complex`.

3.20.2.5 `template<typename _Tp> complex<_Tp> std::cosh (const complex<_Tp> &__z) [inline]`

Return complex hyperbolic cosine of z .

Definition at line 739 of file `complex`.

3.20.2.6 `template<typename _Tp> complex<_Tp> std::exp (const complex<_Tp> &__z) [inline]`

Return complex base e exponential of z .

Definition at line 765 of file `complex`.

Referenced by `std::normal_distribution<_RealType>::operator()()`, `std::lognormal_distribution<_RealType>::operator()()`, `std::negative_binomial_distribution<_IntType>::operator()()`, and `std::poisson_distribution<_IntType>::operator()()`.

3.20.2.7 `template<typename _Tp> std::complex<_Tp> std::tr1::fabs (const std::complex<_Tp> &__z) [inline]`

`fabs(__z)` [8.1.8].

Definition at line 309 of file `tr1/complex`.

3.20.2.8 `template<typename _Tp> complex<_Tp> std::log (const complex<_Tp> &__z) [inline]`

Return complex natural logarithm of z .

Definition at line 792 of file `complex`.

Referenced by `std::generate_canonical()`, `std::normal_distribution<_RealType>::operator()()`, `std::gamma_distribution<_RealType>::operator()()`, `std::binomial_distribution<_IntType>::operator()()`, `std::negative_binomial_distribution<_IntType>::operator()()`, `std::poisson_distribution<_IntType>::operator()()`, `std::exponential_distribution<_RealType>::operator()()`, `std::operator<<()`, and `std::operator>>()`.

3.20.2.9 `template<typename _Tp> complex<_Tp> std::log10 (const complex<_Tp> &__z) [inline]`

Return complex base 10 logarithm of z .

Definition at line 797 of file `complex`.

3.20.2.10 `template<typename _Tp> _Tp std::norm (const complex<_Tp> &__z) [inline]`

Return z magnitude squared.

Definition at line 661 of file `complex`.

3.20.2.11 `template<typename _Tp> constexpr bool std::operator!= (const complex<_Tp> &__x, const complex<_Tp> &__y) [inline]`

Return false if x is equal to y .

Definition at line 476 of file `complex`.

3.20.2.12 `template<typename _Tp> constexpr bool std::operator!= (const complex<_Tp> &__x, const _Tp &__y) [inline]`

Return false if x is equal to y .

Definition at line 481 of file `complex`.

3.20.2.13 `template<typename _Tp> constexpr bool std::operator!= (const _Tp &__x, const complex<_Tp> &__y) [inline]`

Return false if x is equal to y .

Definition at line 486 of file `complex`.

3.20.2.14 `template<typename _Tp> complex<_Tp> std::operator* (const complex<_Tp> &__x, const complex<_Tp> &__y) [inline]`

Return new complex value x times y .

Definition at line 386 of file `complex`.

3.20.2.15 `template<typename _Tp> complex<_Tp> std::operator* (const complex<_Tp> &__x, const _Tp &__y) [inline]`

Return new complex value x times y .

Definition at line 395 of file `complex`.

3.20.2.16 `template<typename _Tp> complex<_Tp> std::operator* (const _Tp &__x, const complex<_Tp> &__y) [inline]`

Return new complex value x times y .

Definition at line 404 of file `complex`.

3.20.2.17 `template<typename _Tp> complex<_Tp> & std::complex<_Tp>::operator*= (const _Tp & __t)`

Multiply this complex number by a scalar.

Definition at line 245 of file complex.

3.20.2.18 `template<typename _Tp> template<typename _Up> complex<_Tp> & std::complex<_Tp>::operator*= (const complex<_Up> & __z)`

Multiply this complex number by another.

Definition at line 299 of file complex.

3.20.2.19 `template<typename _Tp> complex<_Tp> std::operator+ (const complex<_Tp> & __x, const complex<_Tp> & __y) [inline]`

Return new complex value x plus y.

Definition at line 326 of file complex.

Referenced by std::operator+().

3.20.2.20 `template<typename _Tp> complex<_Tp> std::operator+ (const complex<_Tp> & __x, const _Tp & __y) [inline]`

Return new complex value x plus y.

Definition at line 335 of file complex.

3.20.2.21 `template<typename _Tp> complex<_Tp> std::operator+ (const _Tp & __x, const complex<_Tp> & __y) [inline]`

Return new complex value x plus y.

Definition at line 344 of file complex.

3.20.2.22 `template<typename _Tp> complex<_Tp> std::operator+ (const complex<_Tp> & __x) [inline]`

Return x.

Definition at line 445 of file complex.

3.20.2.23 `template<typename _Tp> template<typename _Up> complex<_Tp> & std::complex<_Tp>::operator+= (const complex<_Up> & __z)`

Add another complex number to this one.

Definition at line 276 of file complex.

3.20.2.24 `template<typename _Tp> complex<_Tp> std::operator- (const complex<_Tp> &__x, const complex<_Tp> &__y) [inline]`

Return new complex value x minus y .

Definition at line 356 of file `complex`.

3.20.2.25 `template<typename _Tp> complex<_Tp> std::operator- (const complex<_Tp> &__x, const _Tp &__y) [inline]`

Return new complex value x minus y .

Definition at line 365 of file `complex`.

3.20.2.26 `template<typename _Tp> complex<_Tp> std::operator- (const _Tp &__x, const complex<_Tp> &__y) [inline]`

Return new complex value x minus y .

Definition at line 374 of file `complex`.

3.20.2.27 `template<typename _Tp> complex<_Tp> std::operator- (const complex<_Tp> &__x) [inline]`

Return complex negation of x .

Definition at line 451 of file `complex`.

3.20.2.28 `template<typename _Tp> template<typename _Up> complex<_Tp> & std::complex<_Tp>::operator-= (const complex<_Up> &__z)`

Subtract another complex number from this one.

Definition at line 287 of file `complex`.

3.20.2.29 `template<typename _Tp> complex<_Tp> std::operator/ (const complex<_Tp> &__x, const complex<_Tp> &__y) [inline]`

Return new complex value x divided by y .

Definition at line 416 of file `complex`.

3.20.2.30 `template<typename _Tp> complex<_Tp> std::operator/ (const complex<_Tp> &__x, const _Tp &__y) [inline]`

Return new complex value x divided by y .

Definition at line 425 of file `complex`.

3.20.2.31 `template<typename _Tp> complex<_Tp> std::operator/ (const _Tp & __x, const complex<_Tp> & __y)
[inline]`

Return new complex value x divided by y .

Definition at line 434 of file `complex`.

3.20.2.32 `template<typename _Tp> complex<_Tp> & std::complex<_Tp>::operator/= (const _Tp & __t)`

Divide this complex number by a scalar.

Definition at line 255 of file `complex`.

3.20.2.33 `template<typename _Tp> template<typename _Up> complex<_Tp> & std::complex<_Tp>::operator/= (const
complex<_Up> & __z)`

Divide this complex number by another.

Definition at line 312 of file `complex`.

3.20.2.34 `template<typename _Tp, typename _CharT, class _Traits> basic_ostream<_CharT, _Traits> & std::operator<< (
basic_ostream<_CharT, _Traits> & __os, const complex<_Tp> & __x)`

Insertion operator for complex values.

Definition at line 526 of file `complex`.

3.20.2.35 `template<typename _Tp> complex<_Tp> & std::complex<_Tp>::operator= (const _Tp & __t)`

Assign a scalar to this complex number.

Definition at line 235 of file `complex`.

3.20.2.36 `template<typename _Tp> template<typename _Up> complex<_Tp> & std::complex<_Tp>::operator= (const
complex<_Up> & __z)`

Assign another complex number to this one.

Definition at line 265 of file `complex`.

3.20.2.37 `template<typename _Tp> constexpr bool std::operator==(const complex<_Tp> & __x, const complex<_Tp> & __y) [inline]`

Return true if x is equal to y .

Definition at line 458 of file `complex`.

3.20.2.38 `template<typename _Tp> constexpr bool std::operator==(const complex<_Tp> &__x, const _Tp &__y)`
`[inline]`

Return true if x is equal to y .

Definition at line 463 of file `complex`.

3.20.2.39 `template<typename _Tp> constexpr bool std::operator==(const _Tp &__x, const complex<_Tp> &__y)`
`[inline]`

Return true if x is equal to y .

Definition at line 468 of file `complex`.

3.20.2.40 `template<typename _Tp, typename _CharT, class _Traits> basic_istream<_CharT, _Traits> &std::operator>> (`
`basic_istream<_CharT, _Traits> &__is, complex<_Tp> &__x)`

Extraction operator for complex values.

Definition at line 493 of file `complex`.

3.20.2.41 `template<typename _Tp> complex<_Tp> std::polar (const _Tp &__rho, const _Tp &__theta = 0)` `[inline]`

Return complex with magnitude ρ and angle θ .

Definition at line 669 of file `complex`.

3.20.2.42 `template<typename _Tp> complex<_Tp> std::pow (const complex<_Tp> &__z, int __n)` `[inline]`

Return x to the y 'th power.

Definition at line 987 of file `complex`.

Referenced by `std::gamma_distribution<_RealType>::operator()()`.

3.20.2.43 `template<typename _Tp> complex<_Tp> std::pow (const complex<_Tp> &__x, const _Tp &__y)`

Return x to the y 'th power.

Definition at line 996 of file `complex`.

3.20.2.44 `template<typename _Tp> complex<_Tp> std::pow (const complex<_Tp> &__x, const complex<_Tp> &`
`__y)` `[inline]`

Return x to the y 'th power.

Definition at line 1035 of file `complex`.

3.20.2.45 `template<typename _Tp> complex<_Tp> std::pow (const _Tp & __x, const complex<_Tp> & __y)`
`[inline]`

Return x to the y 'th power.

Definition at line 1041 of file `complex`.

3.20.2.46 `template<typename _Tp, typename _Up> std::complex<typename __gnu_cxx::__promote_2<_Tp, _Up>::__type>`
`std::tr1::pow (const std::complex<_Tp> & __x, const _Up & __y) [inline]`

Additional overloads [8.1.9].

Definition at line 350 of file `tr1/complex`.

3.20.2.47 `template<typename _Tp> complex<_Tp> std::sin (const complex<_Tp> & __z) [inline]`

Return complex sine of z .

Definition at line 827 of file `complex`.

3.20.2.48 `template<typename _Tp> complex<_Tp> std::sinh (const complex<_Tp> & __z) [inline]`

Return complex hyperbolic sine of z .

Definition at line 857 of file `complex`.

3.20.2.49 `template<typename _Tp> complex<_Tp> std::sqrt (const complex<_Tp> & __z) [inline]`

Return complex square root of z .

Definition at line 901 of file `complex`.

Referenced by `std::normal_distribution<_RealType>::operator()()`, `std::student_t_distribution<_RealType>::operator()()`, `std::negative_binomial_distribution<_IntType>::operator()()`, `std::poisson_distribution<_IntType>::operator()()`, and `std::operator>>()`.

3.20.2.50 `template<typename _Tp> complex<_Tp> std::tan (const complex<_Tp> & __z) [inline]`

Return complex tangent of z .

Definition at line 928 of file `complex`.

Referenced by `std::normal_distribution<_RealType>::operator()()`.

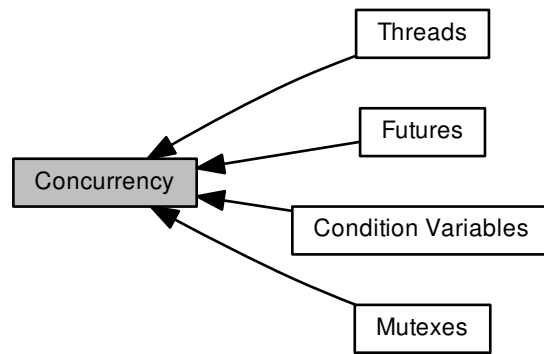
3.20.2.51 `template<typename _Tp> complex<_Tp> std::tanh (const complex<_Tp> & __z) [inline]`

Return complex hyperbolic tangent of z .

Definition at line 956 of file `complex`.

3.21 Concurrency

Collaboration diagram for Concurrency:



Modules

- [Condition Variables](#)
- [Futures](#)
- [Mutexes](#)
- [Threads](#)

3.21.1 Detailed Description

Components for concurrent operations, including threads, mutexes, and condition variables.

3.22 Condition Variables

Collaboration diagram for Condition Variables:



Classes

- class `std::condition_variable`

Enumerations

- enum `std::cv_status` { `no_timeout`, `timeout` }

Functions

- void `std::notify_all_at_thread_exit` (condition_variable &, unique_lock< mutex >)

3.22.1 Detailed Description

Classes for condition_variable support.

3.22.2 Enumeration Type Documentation

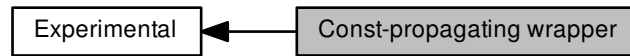
3.22.2.1 enum `std::cv_status` [strong]

`cv_status`

Definition at line 62 of file condition_variable.

3.23 Const-propagating wrapper

Collaboration diagram for Const-propagating wrapper:



Classes

- class `std::experimental::fundamentals_v2::propagate_const< _Tp >`

Functions

- `template<typename _Tp >`
`constexpr const _Tp & std::experimental::fundamentals_v2::get_underlying (const propagate_const< _Tp >`
`&__pt) noexcept`
- `template<typename _Tp >`
`constexpr _Tp & std::experimental::fundamentals_v2::get_underlying (propagate_const< _Tp > &__pt)`
`noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v2::operator!= (const propagate_const< _Tp > &__pt,`
`nullptr_t)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v2::operator!= (nullptr_t, const propagate_const< _Tp >`
`&__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator!= (const propagate_const< _Tp > &__pt, const`
`propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator!= (const propagate_const< _Tp > &__pt, const`
`_Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator!= (const _Tp &__t, const propagate_const< ↵`
`_Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator< (const propagate_const< _Tp > &__pt, const`
`propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator< (const propagate_const< _Tp > &__pt, const`
`_Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator< (const _Tp &__t, const propagate_const< ↵`
`_Up > &__pu)`

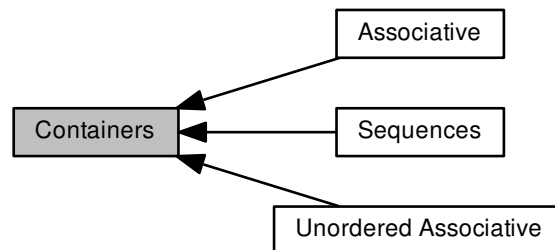
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator<= (const propagate_const< _Tp > &__pt,`
`const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator<= (const propagate_const< _Tp > &__pt,`
`const _Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator<= (const _Tp &__t, const propagate_const<`
`_Up > &__pu)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v2::operator== (const propagate_const< _Tp > &__pt,`
`nullptr_t)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v2::operator== (nullptr_t, const propagate_const< _Tp >`
`&__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator== (const propagate_const< _Tp > &__pt,`
`const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator== (const propagate_const< _Tp > &__pt,`
`const _Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator== (const _Tp &__t, const propagate_const<`
`_Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator> (const propagate_const< _Tp > &__pt, const`
`propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator> (const propagate_const< _Tp > &__pt, const`
`_Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator> (const _Tp &__t, const propagate_const< ↵`
`_Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator>= (const propagate_const< _Tp > &__pt,`
`const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator>= (const propagate_const< _Tp > &__pt,`
`const _Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator>= (const _Tp &__t, const propagate_const<`
`_Up > &__pu)`
- `template<typename _Tp >`
`constexpr void std::experimental::fundamentals_v2::swap (propagate_const< _Tp > &__pt, propagate_↵`
`const< _Tp > &__pt2) noexcept(__is_nothrow_swappable< _Tp >::value)`

3.23.1 Detailed Description

A const-propagating wrapper that propagates const to pointer-like members, as described in n4388 "A Proposal to Add a Const-Propagating Wrapper to the Standard Library".

3.24 Containers

Collaboration diagram for Containers:



Modules

- [Associative](#)
- [Sequences](#)
- [Unordered Associative](#)

3.24.1 Detailed Description

Containers are collections of objects.

A container may hold any type which meets certain requirements, but the type of contained object is chosen at compile time, and all objects in a given container must be of the same type. (Polymorphism is possible by declaring a container of pointers to a base class and then populating it with pointers to instances of derived classes. Variant value types such as the `any` class from `Boost` can also be used.

All contained types must be `Assignable` and `CopyConstructible`. Specific containers may place additional requirements on the types of their contained objects.

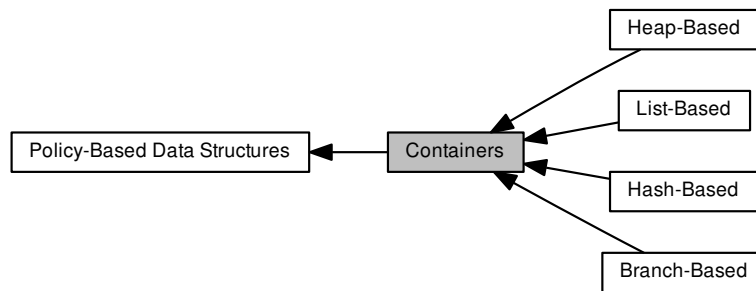
Containers manage memory allocation and deallocation themselves when storing your objects. The objects are destroyed when the container is itself destroyed. Note that if you are storing pointers in a container, `delete` is *not* automatically called on the pointers before destroying them.

All containers must meet certain requirements, summarized in `tables`.

The standard containers are further refined into [Sequences](#) and [Associative Containers](#). [Unordered Associative Containers](#).

3.25 Containers

Collaboration diagram for Containers:



Modules

- [Branch-Based](#)
- [Hash-Based](#)
- [Heap-Based](#)
- [List-Based](#)

3.25.1 Detailed Description

3.26 Data Structure Type

Collaboration diagram for Data Structure Type:



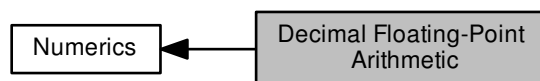
Classes

- struct [__gnu_pbds::associative_tag](#)
- struct [__gnu_pbds::basic_branch_tag](#)
- struct [__gnu_pbds::basic_hash_tag](#)
- struct [__gnu_pbds::binary_heap_tag](#)
- struct [__gnu_pbds::binomial_heap_tag](#)
- struct [__gnu_pbds::cc_hash_tag](#)
- struct [__gnu_pbds::container_tag](#)
- struct [__gnu_pbds::gp_hash_tag](#)
- struct [__gnu_pbds::list_update_tag](#)
- struct [__gnu_pbds::ov_tree_tag](#)
- struct [__gnu_pbds::pairing_heap_tag](#)
- struct [__gnu_pbds::pat_trie_tag](#)
- struct [__gnu_pbds::priority_queue_tag](#)
- struct [__gnu_pbds::rb_tree_tag](#)
- struct [__gnu_pbds::rc_binomial_heap_tag](#)
- struct [__gnu_pbds::sequence_tag](#)
- struct [__gnu_pbds::splay_tree_tag](#)
- struct [__gnu_pbds::string_tag](#)
- struct [__gnu_pbds::thin_heap_tag](#)
- struct [__gnu_pbds::tree_tag](#)
- struct [__gnu_pbds::trie_tag](#)

3.26.1 Detailed Description

3.27 Decimal Floating-Point Arithmetic

Collaboration diagram for Decimal Floating-Point Arithmetic:



Namespaces

- [std::decimal](#)

3.27.1 Detailed Description

Classes and functions for decimal floating-point arithmetic.

3.28 Diagnostics

Collaboration diagram for Diagnostics:



Modules

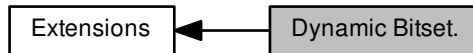
- [Exceptions](#)

3.28.1 Detailed Description

Components for error handling, reporting, and diagnostic operations.

3.29 Dynamic Bitset.

Collaboration diagram for Dynamic Bitset.:



Classes

- struct `std::tr2::__dynamic_bitset_base< _WordT, _Alloc >`
- class `std::tr2::dynamic_bitset< _WordT, _Alloc >`

Functions

- `template<typename _CharT, typename _Traits, typename _Alloc1 >`
`void std::tr2::dynamic_bitset< _WordT, _Alloc >::M_copy_to_string (std::basic_string< _CharT, _Traits, ↵`
`_Alloc1 > &__str, _CharT __zero= _CharT('0'), _CharT __one= _CharT('1')) const`
- `template<typename _CharT, typename _Traits, typename _WordT, typename _Alloc >`
`std::basic_ostream< _CharT, _Traits > & std::tr2::operator<< (std::basic_ostream< _CharT, _Traits > &__os,`
`const dynamic_bitset< _WordT, _Alloc > &__x)`
- `template<typename _CharT, typename _Traits, typename _WordT, typename _Alloc >`
`std::basic_istream< _CharT, _Traits > & std::tr2::operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`dynamic_bitset< _WordT, _Alloc > &__x)`
- `template<typename _WordT, typename _Alloc >`
`bool std::tr2::operator!= (const dynamic_bitset< _WordT, _Alloc > &__lhs, const dynamic_bitset< _WordT, _Alloc`
`> &__rhs)`
- `template<typename _WordT, typename _Alloc >`
`bool std::tr2::operator<= (const dynamic_bitset< _WordT, _Alloc > &__lhs, const dynamic_bitset< _WordT, ↵`
`_Alloc > &__rhs)`
- `template<typename _WordT, typename _Alloc >`
`bool std::tr2::operator> (const dynamic_bitset< _WordT, _Alloc > &__lhs, const dynamic_bitset< _WordT, _Alloc`
`> &__rhs)`
- `template<typename _WordT, typename _Alloc >`
`bool std::tr2::operator>= (const dynamic_bitset< _WordT, _Alloc > &__lhs, const dynamic_bitset< _WordT, ↵`
`_Alloc > &__rhs)`
- `template<typename _WordT, typename _Alloc >`
`dynamic_bitset< _WordT, _Alloc > std::tr2::operator& (const dynamic_bitset< _WordT, _Alloc > &__x, const`
`dynamic_bitset< _WordT, _Alloc > &__y)`

- `template<typename _WordT, typename _Alloc >`
`dynamic_bitset< _WordT, _Alloc > std::tr2::operator| (const dynamic_bitset< _WordT, _Alloc > &__x, const`
`dynamic_bitset< _WordT, _Alloc > &__y)`
- `template<typename _WordT, typename _Alloc >`
`dynamic_bitset< _WordT, _Alloc > std::tr2::operator^ (const dynamic_bitset< _WordT, _Alloc > &__x, const`
`dynamic_bitset< _WordT, _Alloc > &__y)`
- `template<typename _WordT, typename _Alloc >`
`dynamic_bitset< _WordT, _Alloc > std::tr2::operator- (const dynamic_bitset< _WordT, _Alloc > &__x, const`
`dynamic_bitset< _WordT, _Alloc > &__y)`

3.29.1 Detailed Description

3.29.2 Function Documentation

3.29.2.1 `template<typename _WordT, typename _Alloc > bool std::tr2::operator!= (const dynamic_bitset< _WordT, _Alloc >`
`& __lhs, const dynamic_bitset< _WordT, _Alloc > & __rhs) [inline]`

These comparisons for equality/inequality are, well, *bitwise*.

Definition at line 1133 of file `dynamic_bitset`.

3.29.2.2 `template<typename _WordT, typename _Alloc > dynamic_bitset< _WordT, _Alloc> std::tr2::operator& (const`
`dynamic_bitset< _WordT, _Alloc > & __x, const dynamic_bitset< _WordT, _Alloc > & __y) [inline]`

Global bitwise operations on bitsets.

Parameters

<code>__x</code>	A bitset.
<code>__y</code>	A bitset of the same size as <code>__x</code> .

Returns

A new bitset.

These should be self-explanatory.

Definition at line 1168 of file `dynamic_bitset`.

3.29.2.3 `template<typename _WordT, typename _Alloc > dynamic_bitset< _WordT, _Alloc> std::tr2::operator- (const`
`dynamic_bitset< _WordT, _Alloc > & __x, const dynamic_bitset< _WordT, _Alloc > & __y) [inline]`

Global bitwise operations on bitsets.

Parameters

$_x$	A bitset.
$_y$	A bitset of the same size as $_x$.

Returns

A new bitset.

These should be self-explanatory.

Definition at line 1198 of file `dynamic_bitset`.

```
3.29.2.4  template<typename _CharT, typename _Traits, typename _WordT, typename _Alloc > std::basic_ostream<_CharT,
        _Traits>& std::tr2::operator<<( std::basic_ostream<_CharT, _Traits> &__os, const dynamic_bitset<_WordT,
        _Alloc> &__x ) [inline]
```

Stream output operator for `dynamic_bitset`.

Definition at line 1211 of file `dynamic_bitset`.

```
3.29.2.5  template<typename _WordT, typename _Alloc > bool std::tr2::operator<=( const dynamic_bitset<_WordT, _Alloc>
        &__lhs, const dynamic_bitset<_WordT, _Alloc> &__rhs ) [inline]
```

These comparisons for equality/inequality are, well, *bitwise*.

Definition at line 1139 of file `dynamic_bitset`.

```
3.29.2.6  template<typename _WordT, typename _Alloc > bool std::tr2::operator>( const dynamic_bitset<_WordT, _Alloc>
        &__lhs, const dynamic_bitset<_WordT, _Alloc> &__rhs ) [inline]
```

These comparisons for equality/inequality are, well, *bitwise*.

Definition at line 1145 of file `dynamic_bitset`.

```
3.29.2.7  template<typename _WordT, typename _Alloc > bool std::tr2::operator>=( const dynamic_bitset<_WordT, _Alloc>
        &__lhs, const dynamic_bitset<_WordT, _Alloc> &__rhs ) [inline]
```

These comparisons for equality/inequality are, well, *bitwise*.

Definition at line 1151 of file `dynamic_bitset`.

```
3.29.2.8 template<typename _CharT, typename _Traits, typename _WordT, typename _Alloc > std::basic_istream<_CharT,
    _Traits>& std::tr2::operator>> ( std::basic_istream<_CharT, _Traits> & __is, dynamic_bitset<_WordT, _Alloc
    > & __x )
```

Stream input operator for dynamic_bitset.

Input will skip whitespace and only accept '0' and '1' characters. The dynamic_bitset will grow as necessary to hold the string of bits.

Definition at line 207 of file dynamic_bitset.tcc.

References std::basic_string<_CharT, _Traits, _Alloc>::empty(), std::basic_string<_CharT, _Traits, _Alloc>::push_back(), std::basic_ios<_CharT, _Traits>::rdbuf(), std::basic_string<_CharT, _Traits, _Alloc>::reserve(), std::tr2::dynamic_bitset<_WordT, _Alloc>::resize(), std::basic_ios<_CharT, _Traits>::setstate(), std::tr2::dynamic_bitset<_WordT, _Alloc>::size(), std::basic_string<_CharT, _Traits, _Alloc>::size(), and std::basic_ios<_CharT, _Traits>::widen().

```
3.29.2.9 template<typename _WordT, typename _Alloc > dynamic_bitset<_WordT, _Alloc> std::tr2::operator^ ( const
    dynamic_bitset<_WordT, _Alloc> & __x, const dynamic_bitset<_WordT, _Alloc> & __y ) [inline]
```

Global bitwise operations on bitsets.

Parameters

$_x$	A bitset.
$_y$	A bitset of the same size as $_x$.

Returns

A new bitset.

These should be self-explanatory.

Definition at line 1188 of file dynamic_bitset.

```
3.29.2.10 template<typename _WordT, typename _Alloc > dynamic_bitset<_WordT, _Alloc> std::tr2::operator| ( const
    dynamic_bitset<_WordT, _Alloc> & __x, const dynamic_bitset<_WordT, _Alloc> & __y ) [inline]
```

Global bitwise operations on bitsets.

Parameters

$_x$	A bitset.
$_y$	A bitset of the same size as $_x$.

Returns

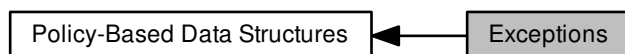
A new bitset.

These should be self-explanatory.

Definition at line 1178 of file dynamic_bitset.

3.30 Exceptions

Collaboration diagram for Exceptions:



Classes

- struct [__gnu_pbds::container_error](#)
- struct [__gnu_pbds::insert_error](#)
- struct [__gnu_pbds::join_error](#)
- struct [__gnu_pbds::resize_error](#)

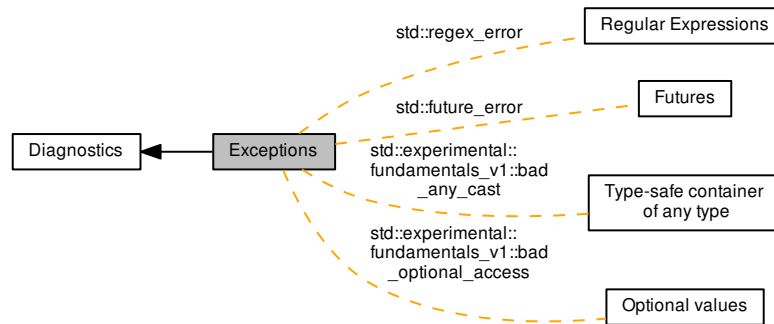
Functions

- void [__gnu_pbds::__throw_container_error\(\)](#)
- void [__gnu_pbds::__throw_insert_error\(\)](#)
- void [__gnu_pbds::__throw_join_error\(\)](#)
- void [__gnu_pbds::__throw_resize_error\(\)](#)

3.30.1 Detailed Description

3.31 Exceptions

Collaboration diagram for Exceptions:



Classes

- class `__cxxabiv1::__forced_unwind`
- struct `__gnu_cxx::forced_error`
- class `__gnu_cxx::recursive_init_error`
- class `std::__exception_ptr::exception_ptr`
- class `std::bad_alloc`
- class `std::bad_cast`
- class `std::bad_exception`
- class `std::bad_function_call`
- class `std::bad_typeid`
- class `std::bad_weak_ptr`
- class `std::domain_error`
- class `std::exception`
- class `std::experimental::fundamentals_v1::bad_any_cast`
- class `std::experimental::fundamentals_v1::bad_optional_access`
- class `std::future_error`
- class `std::invalid_argument`
- class `std::ios_base::failure`
- class `std::length_error`
- class `std::logic_error`
- class `std::nested_exception`
- class `std::out_of_range`
- class `std::overflow_error`
- class `std::range_error`
- class `std::regex_error`
- class `std::runtime_error`
- class `std::system_error`
- class `std::underflow_error`

Macros

- `#define __cpp_lib_uncaught_exceptions`

Typedefs

- `template<typename _Tp >`
`using std::__rethrow_if_nested_cond = typename enable_if< __and_< is_polymorphic< _Tp >, __or_< __not_< is_base_of< nested_exception, _Tp >>, is_convertible< _Tp *, nested_exception * >>>::value >::type`
- `typedef void(* std::terminate_handler) ()`
- `typedef void(* std::unexpected_handler) ()`

Functions

- `template<typename _Ex >`
`__rethrow_if_nested_cond< _Ex > std::__rethrow_if_nested_impl (const _Ex *__ptr)`
- `void std::__rethrow_if_nested_impl (const void *)`
- `template<typename _Tp >`
`void std::__throw_with_nested_impl (_Tp &&__t, true_type)`
- `template<typename _Tp >`
`void std::__throw_with_nested_impl (_Tp &&__t, false_type)`
- `void __gnu_cxx::__verbose_terminate_handler ()`
- `template<typename _Ex >`
`exception_ptr std::copy_exception (_Ex __ex) noexcept 1`
- `exception_ptr std::current_exception () noexcept`
- `terminate_handler std::get_terminate () noexcept`
- `unexpected_handler std::get_unexpected () noexcept`
- `template<typename _Ex >`
`exception_ptr std::make_exception_ptr (_Ex __ex) noexcept`
- `void std::rethrow_exception (exception_ptr) __attribute__((__noreturn__))`
- `template<typename _Ex >`
`void std::rethrow_if_nested (const _Ex &__ex)`
- `terminate_handler std::set_terminate (terminate_handler) noexcept`
- `unexpected_handler std::set_unexpected (unexpected_handler) noexcept`
- `void std::terminate () noexcept __attribute__((__noreturn__))`
- `template<typename _Tp >`
`void std::throw_with_nested (_Tp &&__t)`
- `bool std::uncaught_exception () noexcept __attribute__((__pure__))`
- `int std::uncaught_exceptions () noexcept __attribute__((__pure__))`
- `void std::unexpected () __attribute__((__noreturn__))`

3.31.1 Detailed Description

Classes and functions for reporting errors via exception classes.

3.31.2 Typedef Documentation

3.31.2.1 `typedef void(* std::terminate_handler) ()`

If you write a replacement terminate handler, it must be of this type.

Definition at line 89 of file exception.

3.31.2.2 `typedef void(* std::unexpected_handler) ()`

If you write a replacement unexpected handler, it must be of this type.

Definition at line 92 of file exception.

3.31.3 Function Documentation

3.31.3.1 `void __gnu_cxx::__verbose_terminate_handler ()`

A replacement for the standard `terminate_handler` which prints more information about the terminating exception (if any) on `stderr`.

Call

```
std::set_terminate(__gnu_cxx::__verbose_terminate_handler
)
```

to use. For more info, see <http://gcc.gnu.org/onlinedocs/libstdc++/manual/bk01pt02ch06s02.html>.

In 3.4 and later, this is on by default.

3.31.3.2 `template<typename _Ex> exception_ptr std::copy_exception (_Ex __ex) [noexcept]`

Obtain an `exception_ptr` pointing to a copy of the supplied object. This function is deprecated, use `std::make_exception_ptr` instead.

Definition at line 197 of file `exception_ptr.h`.

Referenced by `std::make_exception_ptr()`.

3.31.3.3 `exception_ptr std::current_exception () [noexcept]`

Obtain an `exception_ptr` to the currently handled exception. If there is none, or the currently handled exception is foreign, return the null value.

Referenced by `std::make_exception_ptr()`.

3.31.3.4 `terminate_handler std::get_terminate () [noexcept]`

Return the current terminate handler.

3.31.3.5 `unexpected_handler std::get_unexpected () [noexcept]`

Return the current unexpected handler.

3.31.3.6 `template<typename _Ex > exception_ptr std::make_exception_ptr (_Ex __ex) [noexcept]`

Obtain an `exception_ptr` pointing to a copy of the supplied object.

Definition at line 171 of file `exception_ptr.h`.

References `std::copy_exception()`, and `std::current_exception()`.

3.31.3.7 `void std::rethrow_exception (exception_ptr)`

Throw the object pointed to by the `exception_ptr`.

3.31.3.8 `template<typename _Ex > void std::rethrow_if_nested (const _Ex & __ex) [inline]`

If `__ex` is derived from `nested_exception`, `__ex.rethrow_nested()`.

Definition at line 153 of file `nested_exception.h`.

References `std::__addressof()`.

3.31.3.9 `terminate_handler std::set_terminate (terminate_handler) [noexcept]`

Takes a new handler function as an argument, returns the old function.

3.31.3.10 `unexpected_handler std::set_unexpected (unexpected_handler) [noexcept]`

Takes a new handler function as an argument, returns the old function.

3.31.3.11 `void std::terminate () [noexcept]`

The runtime will call this function if exception handling must be abandoned for any reason. It can also be called by the user.

3.31.3.12 `template<typename _Tp > void std::throw_with_nested (_Tp && __t) [inline]`

If `__t` is derived from `nested_exception`, throws `__t`. Else, throws an implementation-defined object derived from both.

Definition at line 116 of file `nested_exception.h`.

3.31.3.13 `bool std::uncaught_exception () [noexcept]`

[18.6.4]/1: 'Returns true after completing evaluation of a throw-expression until either completing initialization of the exception-declaration in the matching handler or entering `unexpected()` due to the throw; or after entering `terminate()` for any reason other than an explicit call to `terminate()`. [Note: This includes stack unwinding [15.2]. end note]'

2: 'When `uncaught_exception()` is true, throwing an exception can result in a call of `terminate()` (15.5.1).'

3.31.3.14 `int std::uncaught_exceptions () [noexcept]`

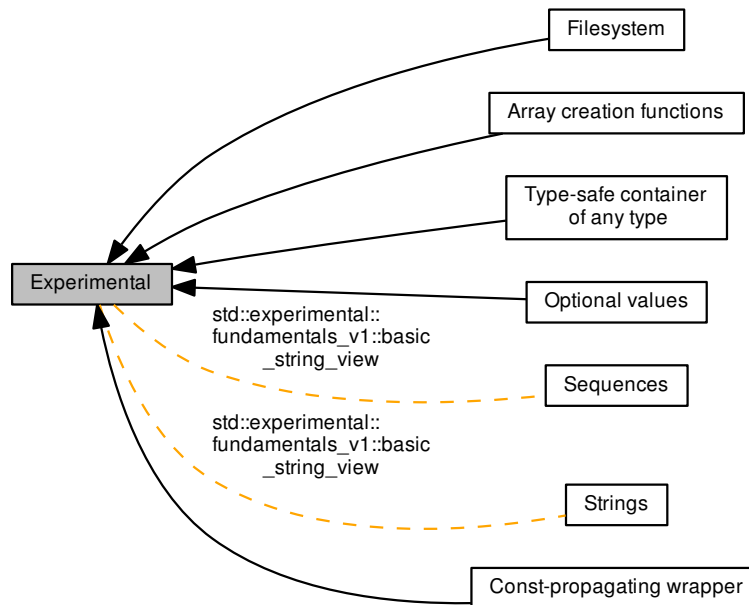
The number of uncaught exceptions.

3.31.3.15 `void std::unexpected ()`

The runtime will call this function if an exception is thrown which violates the function's exception specification.

3.32 Experimental

Collaboration diagram for Experimental:



Modules

- [Array creation functions](#)
- [Const-propagating wrapper](#)
- [Filesystem](#)
- [Optional values](#)
- [Type-safe container of any type](#)

Classes

- class [std::experimental::fundamentals_v1::basic_string_view<_CharT, _Traits>](#)

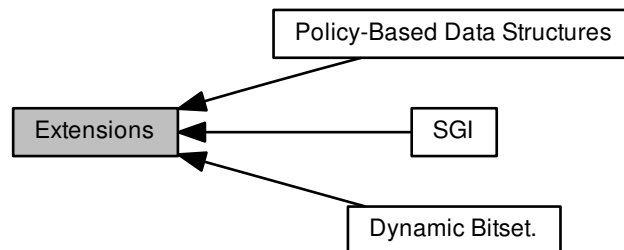
3.32.1 Detailed Description

Components specified by various Technical Specifications.

As indicated by the `std::experimental` namespace and the header paths, the contents of these Technical Specifications are experimental and not part of the C++ standard. As such the interfaces and implementations may change in the future, and there is **no guarantee of compatibility between different GCC releases** for these features.

3.33 Extensions

Collaboration diagram for Extensions:



Modules

- [Dynamic Bitset.](#)
- [Policy-Based Data Structures](#)
- [SGI](#)

Classes

- [class `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>`](#)

3.33.1 Detailed Description

Components generally useful that are not part of any standard.

3.34 Filesystem

Collaboration diagram for Filesystem:



Classes

- class `std::experimental::filesystem::v1::path`

Typedefs

- typedef `chrono::time_point< chrono::system_clock > std::experimental::filesystem::v1::file_time_type`

Enumerations

- enum `std::experimental::filesystem::v1::copy_options` : unsigned short { **none**, **skip_existing**, **overwrite_existing**, **update_existing**, **recursive**, **copy_symlinks**, **skip_symlinks**, **directories_only**, **create_symlinks**, **create_hard_links** }
- enum `directory_options` : unsigned char { **none**, **follow_directory_symlink**, **skip_permission_denied** }
- enum `file_type` : signed char { **none**, **not_found**, **regular**, **directory**, **symlink**, **block**, **character**, **fifo**, **socket**, **unknown** }
- enum `std::experimental::filesystem::v1::perms` : unsigned { **none**, **owner_read**, **owner_write**, **owner_exec**, **owner_all**, **group_read**, **group_write**, **group_exec**, **group_all**, **others_read**, **others_write**, **others_exec**, **others_all**, **all**, **set_uid**, **set_gid**, **sticky_bit**, **mask**, **unknown**, **add_perms**, **remove_perms**, **symlink_nofollow** }

Functions

- constexpr copy_options **std::experimental::filesystem::v1::operator&** (copy_options __x, copy_options __y) noexcept
 - constexpr perms **std::experimental::filesystem::v1::operator&** (perms __x, perms __y) noexcept
 - constexpr directory_options **std::experimental::filesystem::v1::operator&** (directory_options __x, directory_options __y) noexcept
 - copy_options & **std::experimental::filesystem::v1::operator&=** (copy_options &__x, copy_options __y) noexcept
 - perms & **std::experimental::filesystem::v1::operator&=** (perms &__x, perms __y) noexcept
 - directory_options & **std::experimental::filesystem::v1::operator&=** (directory_options &__x, directory_options __y) noexcept
 - constexpr copy_options **std::experimental::filesystem::v1::operator^** (copy_options __x, copy_options __y) noexcept
 - constexpr perms **std::experimental::filesystem::v1::operator^** (perms __x, perms __y) noexcept
 - constexpr directory_options **std::experimental::filesystem::v1::operator^** (directory_options __x, directory_options __y) noexcept
 - copy_options & **std::experimental::filesystem::v1::operator^=** (copy_options &__x, copy_options __y) noexcept
 - perms & **std::experimental::filesystem::v1::operator^=** (perms &__x, perms __y) noexcept
 - directory_options & **std::experimental::filesystem::v1::operator^=** (directory_options &__x, directory_options __y) noexcept
 - constexpr copy_options **std::experimental::filesystem::v1::operator|** (copy_options __x, copy_options __y) noexcept
 - constexpr perms **std::experimental::filesystem::v1::operator|** (perms __x, perms __y) noexcept
 - constexpr directory_options **std::experimental::filesystem::v1::operator|** (directory_options __x, directory_options __y) noexcept
 - copy_options & **std::experimental::filesystem::v1::operator|=** (copy_options &__x, copy_options __y) noexcept
 - perms & **std::experimental::filesystem::v1::operator|=** (perms &__x, perms __y) noexcept
 - directory_options & **std::experimental::filesystem::v1::operator|=** (directory_options &__x, directory_options __y) noexcept
 - constexpr copy_options **std::experimental::filesystem::v1::operator~** (copy_options __x) noexcept
 - constexpr perms **std::experimental::filesystem::v1::operator~** (perms __x) noexcept
 - constexpr directory_options **std::experimental::filesystem::v1::operator~** (directory_options __x) noexcept
-
- void **std::experimental::filesystem::v1::copy** (const path &__from, const path &__to, copy_options __options)
 - void **std::experimental::filesystem::v1::copy** (const path &__from, const path &__to, copy_options __options, error_code &) noexcept
 - bool **std::experimental::filesystem::v1::copy_file** (const path &__from, const path &__to, copy_options __option)
 - bool **std::experimental::filesystem::v1::copy_file** (const path &__from, const path &__to, copy_options __option, error_code &) noexcept
 - path **std::experimental::filesystem::v1::current_path** ()
 - file_status **std::experimental::filesystem::v1::status** (const path &)
 - file_status **std::experimental::filesystem::v1::status** (const path &, error_code &) noexcept
 - bool **std::experimental::filesystem::v1::status_known** (file_status) noexcept
 - file_status **std::experimental::filesystem::v1::symlink_status** (const path &)
 - file_status **std::experimental::filesystem::v1::symlink_status** (const path &, error_code &) noexcept
 - bool **std::experimental::filesystem::v1::is_regular_file** (file_status) noexcept
 - bool **std::experimental::filesystem::v1::is_symlink** (file_status) noexcept

- path **std::experimental::filesystem::v1::absolute** (const path &__p, const path &__base=current_path())
- path **std::experimental::filesystem::v1::canonical** (const path &__p, const path &__base=current_path())
- path **std::experimental::filesystem::v1::canonical** (const path &__p, error_code &__ec)
- path **std::experimental::filesystem::v1::canonical** (const path &__p, const path &__base, error_code &__ec)
- void **std::experimental::filesystem::v1::copy** (const path &__from, const path &__to)
- void **std::experimental::filesystem::v1::copy** (const path &__from, const path &__to, error_code &__ec) noexcept
- bool **std::experimental::filesystem::v1::copy_file** (const path &__from, const path &__to)
- bool **std::experimental::filesystem::v1::copy_file** (const path &__from, const path &__to, error_code &__ec) noexcept
- void **std::experimental::filesystem::v1::copy_symlink** (const path &__existing_symlink, const path &__new_↵
_symlink)
- void **std::experimental::filesystem::v1::copy_symlink** (const path &__existing_symlink, const path &__new_↵
_symlink, error_code &__ec) noexcept
- bool **std::experimental::filesystem::v1::create_directories** (const path &__p)
- bool **std::experimental::filesystem::v1::create_directories** (const path &__p, error_code &__ec) noexcept
- bool **std::experimental::filesystem::v1::create_directory** (const path &__p)
- bool **std::experimental::filesystem::v1::create_directory** (const path &__p, error_code &__ec) noexcept
- bool **std::experimental::filesystem::v1::create_directory** (const path &__p, const path &attributes)
- bool **std::experimental::filesystem::v1::create_directory** (const path &__p, const path &attributes, error_code &__ec) noexcept
- void **std::experimental::filesystem::v1::create_directory_symlink** (const path &__to, const path &__new_↵
symlink)
- void **std::experimental::filesystem::v1::create_directory_symlink** (const path &__to, const path &__new_↵
symlink, error_code &__ec) noexcept
- void **std::experimental::filesystem::v1::create_hard_link** (const path &__to, const path &__new_hard_link)
- void **std::experimental::filesystem::v1::create_hard_link** (const path &__to, const path &__new_hard_link, error_code &__ec) noexcept
- void **std::experimental::filesystem::v1::create_symlink** (const path &__to, const path &__new_symlink)
- void **std::experimental::filesystem::v1::create_symlink** (const path &__to, const path &__new_symlink, error_code &__ec) noexcept
- path **std::experimental::filesystem::v1::current_path** (error_code &__ec)
- void **std::experimental::filesystem::v1::current_path** (const path &__p)
- void **std::experimental::filesystem::v1::current_path** (const path &__p, error_code &__ec) noexcept
- bool **std::experimental::filesystem::v1::equivalent** (const path &__p1, const path &__p2)
- bool **std::experimental::filesystem::v1::equivalent** (const path &__p1, const path &__p2, error_code &__ec) noexcept
- bool **std::experimental::filesystem::v1::exists** (file_status __s) noexcept
- bool **std::experimental::filesystem::v1::exists** (const path &__p)
- bool **std::experimental::filesystem::v1::exists** (const path &__p, error_code &__ec) noexcept
- uintmax_t **std::experimental::filesystem::v1::file_size** (const path &__p)
- uintmax_t **std::experimental::filesystem::v1::file_size** (const path &__p, error_code &__ec) noexcept
- uintmax_t **std::experimental::filesystem::v1::hard_link_count** (const path &__p)
- uintmax_t **std::experimental::filesystem::v1::hard_link_count** (const path &__p, error_code &__ec) noexcept
- bool **std::experimental::filesystem::v1::is_block_file** (file_status __s) noexcept
- bool **std::experimental::filesystem::v1::is_block_file** (const path &__p)
- bool **std::experimental::filesystem::v1::is_block_file** (const path &__p, error_code &__ec) noexcept
- bool **std::experimental::filesystem::v1::is_character_file** (file_status __s) noexcept
- bool **std::experimental::filesystem::v1::is_character_file** (const path &__p)
- bool **std::experimental::filesystem::v1::is_character_file** (const path &__p, error_code &__ec) noexcept
- bool **std::experimental::filesystem::v1::is_directory** (file_status __s) noexcept

- `bool std::experimental::filesystem::v1::is_directory (const path &__p)`
- `bool std::experimental::filesystem::v1::is_directory (const path &__p, error_code &__ec) noexcept`
- `bool std::experimental::filesystem::v1::is_empty (const path &__p)`
- `bool std::experimental::filesystem::v1::is_empty (const path &__p, error_code &__ec) noexcept`
- `bool std::experimental::filesystem::v1::is_fifo (file_status __s) noexcept`
- `bool std::experimental::filesystem::v1::is_fifo (const path &__p)`
- `bool std::experimental::filesystem::v1::is_fifo (const path &__p, error_code &__ec) noexcept`
- `bool std::experimental::filesystem::v1::is_other (file_status __s) noexcept`
- `bool std::experimental::filesystem::v1::is_other (const path &__p)`
- `bool std::experimental::filesystem::v1::is_other (const path &__p, error_code &__ec) noexcept`
- `bool std::experimental::filesystem::v1::is_regular_file (const path &__p)`
- `bool std::experimental::filesystem::v1::is_regular_file (const path &__p, error_code &__ec) noexcept`
- `bool std::experimental::filesystem::v1::is_socket (file_status __s) noexcept`
- `bool std::experimental::filesystem::v1::is_socket (const path &__p)`
- `bool std::experimental::filesystem::v1::is_socket (const path &__p, error_code &__ec) noexcept`
- `bool std::experimental::filesystem::v1::is_symlink (const path &__p)`
- `bool std::experimental::filesystem::v1::is_symlink (const path &__p, error_code &__ec) noexcept`
- `file_time_type std::experimental::filesystem::v1::last_write_time (const path &__p)`
- `file_time_type std::experimental::filesystem::v1::last_write_time (const path &__p, error_code &__ec) noexcept`
- `void std::experimental::filesystem::v1::last_write_time (const path &__p, file_time_type __new_time)`
- `void std::experimental::filesystem::v1::last_write_time (const path &__p, file_time_type __new_time, error_code &__ec) noexcept`
- `void std::experimental::filesystem::v1::permissions (const path &__p, perms __prms)`
- `void std::experimental::filesystem::v1::permissions (const path &__p, perms __prms, error_code &__ec) noexcept`
- `path std::experimental::filesystem::v1::read_symlink (const path &__p)`
- `path std::experimental::filesystem::v1::read_symlink (const path &__p, error_code &__ec)`
- `bool std::experimental::filesystem::v1::remove (const path &__p)`
- `bool std::experimental::filesystem::v1::remove (const path &__p, error_code &__ec) noexcept`
- `uintmax_t std::experimental::filesystem::v1::remove_all (const path &__p)`
- `uintmax_t std::experimental::filesystem::v1::remove_all (const path &__p, error_code &__ec) noexcept`
- `void std::experimental::filesystem::v1::rename (const path &__from, const path &__to)`
- `void std::experimental::filesystem::v1::rename (const path &__from, const path &__to, error_code &__ec) noexcept`
- `void std::experimental::filesystem::v1::resize_file (const path &__p, uintmax_t __size)`
- `void std::experimental::filesystem::v1::resize_file (const path &__p, uintmax_t __size, error_code &__ec) noexcept`
- `space_info std::experimental::filesystem::v1::space (const path &__p)`
- `space_info std::experimental::filesystem::v1::space (const path &__p, error_code &__ec) noexcept`
- `path std::experimental::filesystem::v1::system_complete (const path &__p)`
- `path std::experimental::filesystem::v1::system_complete (const path &__p, error_code &__ec)`
- `path std::experimental::filesystem::v1::temp_directory_path ()`
- `path std::experimental::filesystem::v1::temp_directory_path (error_code &__ec)`

3.34.1 Detailed Description

Utilities for performing operations on file systems and their components, such as paths, regular files, and directories.

3.34.2 Enumeration Type Documentation

3.34.2.1 `enum std::experimental::filesystem::v1::copy_options : unsigned short` `[strong]`

Bitmask type.

Definition at line 90 of file `fs_fwd.h`.

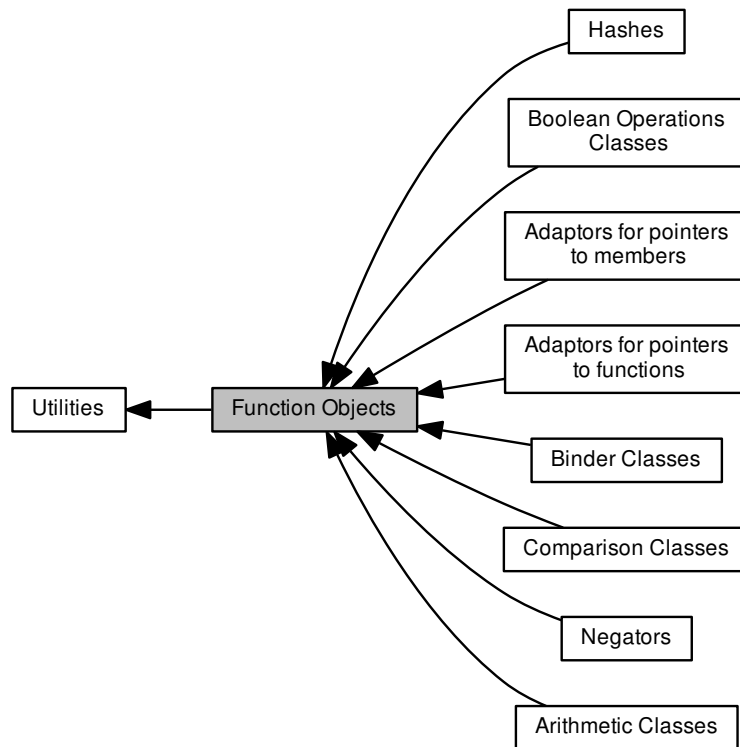
3.34.2.2 `enum std::experimental::filesystem::v1::perms : unsigned` `[strong]`

Bitmask type.

Definition at line 143 of file `fs_fwd.h`.

3.35 Function Objects

Collaboration diagram for Function Objects:



Modules

- [Adaptors for pointers to functions](#)
- [Adaptors for pointers to members](#)
- [Arithmetic Classes](#)
- [Binder Classes](#)
- [Boolean Operations Classes](#)
- [Comparison Classes](#)
- [Hashes](#)
- [Negators](#)

Classes

- `struct std::binary_function< _Arg1, _Arg2, _Result >`
- `class std::function< _Res(_ArgTypes...)>`
- `class std::reference_wrapper< _Tp >`
- `struct std::unary_function< _Arg, _Result >`

Functions

- `template<typename _Tp, typename _Class >
_Mem_fn< _Tp _Class::* > std::mem_fn (_Tp _Class::* __pm) noexcept`

3.35.1 Detailed Description

Function objects, or *functors*, are objects with an `operator()` defined and accessible. They can be passed as arguments to algorithm templates and used in place of a function pointer. Not only is the resulting expressiveness of the library increased, but the generated code can be more efficient than what you might write by hand. When we refer to *functors*, then, generally we include function pointers in the description as well.

Often, functors are only created as temporaries passed to algorithm calls, rather than being created as named variables.

Two examples taken from the standard itself follow. To perform a by-element addition of two vectors `a` and `b` containing `double`, and put the result in `a`, use

```
transform (a.begin(), a.end(), b.begin(), a.begin(), plus<double>());
```

To negate every element in `a`, use

```
transform(a.begin(), a.end(), a.begin(), negate<double>());
```

The addition and negation functions will be inlined directly.

The standard functors are derived from structs named `unary_function` and `binary_function`. These two classes contain nothing but typedefs, to aid in generic (template) programming. If you write your own functors, you might consider doing the same.

3.35.2 Function Documentation

3.35.2.1 `template<typename _Tp, typename _Class > _Mem_fn< _Tp _Class::* > std::mem_fn (_Tp _Class::* __pm) [inline], [noexcept]`

Returns a function object that forwards to the member pointer `pm`.

Definition at line 647 of file `functional`.

3.36 Futures

Collaboration diagram for Futures:



Classes

- class `std::__basic_future< _Res >`
- struct `std::__future_base`
- struct `std::__future_base::Result< _Res & >`
- struct `std::__future_base::Result< void >`
- class `std::future< _Res >`
- class `std::future< _Res & >`
- class `std::future< void >`
- class `std::future_error`
- struct `std::is_error_code_enum< future_errc >`
- class `std::packaged_task< _Res(_ArgTypes...)>`
- class `std::promise< _Res >`
- class `std::promise< _Res & >`
- class `std::promise< void >`
- class `std::shared_future< _Res >`
- class `std::shared_future< _Res & >`
- class `std::shared_future< void >`

Typedefs

- template<typename _Fn , typename... _Args>
using `std::__async_result_of` = typename result_of< typename decay< _Fn >::type(typename decay< _Args >::type...)>::type

Enumerations

- enum `std::future_errc` { `future_already_retrieved`, `promise_already_satisfied`, `no_state`, `broken_promise` }
- enum `std::future_status` { `ready`, `timeout`, `deferred` }
- enum `std::launch` { `async`, `deferred` }

Functions

- **std::__basic_future< _Res >::__basic_future** (const shared_future< _Res > &) noexcept
- **std::__basic_future< _Res >::__basic_future** (shared_future< _Res > &&) noexcept
- **std::__basic_future< _Res >::__basic_future** (future< _Res > &&) noexcept
- template<typename _Signature, typename _Fn, typename _Alloc >
static shared_ptr< __future_base::__Task_state_base< _Signature > > **std::__create_task_state** (_Fn &&__fn, const _Alloc &__a)
- template<typename _BoundFn >
static **std::shared_ptr**< _State_base > **std::__future_base::S_make_async_state** (_BoundFn &&__fn)
- template<typename _BoundFn >
static **std::shared_ptr**< _State_base > **std::__future_base::S_make_deferred_state** (_BoundFn &&__fn)
- template<typename _Fn, typename... _Args>
future< __async_result_of< _Fn, _Args... > > **std::async** (launch __policy, _Fn &&__fn, _Args &&...__args)
- template<typename _Fn, typename... _Args>
future< __async_result_of< _Fn, _Args... > > **std::async** (_Fn &&__fn, _Args &&...__args)
- const error_category & **std::future_category** () noexcept
- error_code **std::make_error_code** (future_errc __errc) noexcept
- error_condition **std::make_error_condition** (future_errc __errc) noexcept
- constexpr launch **std::operator&** (launch __x, launch __y)
- launch & **std::operator&=** (launch &__x, launch __y)
- constexpr launch **std::operator^** (launch __x, launch __y)
- launch & **std::operator^=** (launch &__x, launch __y)
- constexpr launch **std::operator|** (launch __x, launch __y)
- launch & **std::operator|=** (launch &__x, launch __y)
- constexpr launch **std::operator~** (launch __x)
- void **std::promise< void >::set_value** ()
- void **std::promise< void >::set_value_at_thread_exit** ()
- shared_future< _Res > **std::future< _Res >::share** ()
- shared_future< _Res & > **std::future< _Res & >::share** ()
- shared_future< void > **std::future< void >::share** ()
- template<typename _Res >
void **std::swap** (promise< _Res > &__x, promise< _Res > &__y) noexcept
- template<typename _Res, typename... _ArgTypes>
void **std::swap** (packaged_task< _Res(_ArgTypes...) > &__x, packaged_task< _Res(_ArgTypes...) > &__y) noexcept

3.36.1 Detailed Description

Classes for futures support.

3.36.2 Enumeration Type Documentation

3.36.2.1 enum **std::future_errc** [strong]

Error code for futures.

Definition at line 65 of file future.

3.36.2.2 `enum std::future_status` `[strong]`

Status code for futures.

Definition at line 164 of file future.

3.36.2.3 `enum std::launch` `[strong]`

Launch code for futures.

Definition at line 127 of file future.

3.36.3 Function Documentation

3.36.3.1 `template<typename _Fn, typename... _Args> future< __async_result_of< _Fn, _Args... > > std::async (launch __policy, _Fn && __fn, _Args &&... __args)`

async

Definition at line 1709 of file future.

3.36.3.2 `template<typename _Fn, typename... _Args> future< __async_result_of< _Fn, _Args... > > std::async (_Fn && __fn, _Args &&... __args)` `[inline]`

async, potential overload

Definition at line 1739 of file future.

3.36.3.3 `const error_category& std::future_category ()` `[noexcept]`

Points to a statically-allocated object derived from error_category.

3.36.3.4 `error_code std::make_error_code (future_errc __errc)` `[inline]`, `[noexcept]`

Overload for make_error_code.

Definition at line 83 of file future.

3.36.3.5 `error_condition std::make_error_condition (future_errc __errc)` `[inline]`, `[noexcept]`

Overload for make_error_condition.

Definition at line 88 of file future.

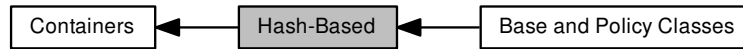
3.36.3.6 `template<typename _Res, typename... _ArgTypes> void std::swap (packaged_task< _Res(_ArgTypes...)> & __x, packaged_task< _Res(_ArgTypes...)> & __y)` `[inline]`, `[noexcept]`

swap

Definition at line 1576 of file future.

3.37 Hash-Based

Collaboration diagram for Hash-Based:



Modules

- [Base and Policy Classes](#)

Classes

- [class `__gnu_pbds::basic_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Resize_Policy, Store_Hash, Tag, Policy_Ti, _Alloc >`](#)
- [class `__gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc >`](#)
- [class `__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >`](#)

Macros

- `#define PB_DS_CC_HASH_BASE`
- `#define PB_DS_GP_HASH_BASE`
- `#define PB_DS_HASH_BASE`

3.37.1 Detailed Description

3.38 Hashes

Collaboration diagram for Hashes:



Classes

- struct `std::hash<_Tp>`
- struct `std::hash<_Tp*>`
- struct `std::hash<bool>`
- struct `std::hash<char>`
- struct `std::hash<char16_t>`
- struct `std::hash<char32_t>`
- struct `std::hash<double>`
- struct `std::hash<float>`
- struct `std::hash<int>`
- struct `std::hash<long>`
- struct `std::hash<long double>`
- struct `std::hash<long long>`
- struct `std::hash<short>`
- struct `std::hash<signed char>`
- struct `std::hash<unsigned char>`
- struct `std::hash<unsigned int>`
- struct `std::hash<unsigned long>`
- struct `std::hash<unsigned long long>`
- struct `std::hash<unsigned short>`
- struct `std::hash<wchar_t>`

Macros

- `#define _Cxx_hashtable_define_trivial_hash(_Tp)`

3.38.1 Detailed Description

Hashing functors taking a variable type and returning a `std::size_t`.

3.39 Heap

Collaboration diagram for Heap:



Functions

- `template<typename _RandomAccessIterator >`
`bool std::is_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator , typename _Compare >`
`bool std::is_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`_RandomAccessIterator std::is_heap_until (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator , typename _Compare >`
`_RandomAccessIterator std::is_heap_until (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`void std::make_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator , typename _Compare >`
`void std::make_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`void std::pop_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator , typename _Compare >`
`void std::pop_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`void std::push_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator , typename _Compare >`
`void std::push_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`void std::sort_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator , typename _Compare >`
`void std::sort_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`

3.39.1 Detailed Description

3.39.2 Function Documentation

3.39.2.1 `template<typename _RandomAccessIterator > bool std::is_heap (_RandomAccessIterator __first, _RandomAccessIterator __last) [inline]`

Determines whether a range is a heap.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.

Returns

True if range is a heap, false otherwise.

Definition at line 519 of file `stl_heap.h`.

References `std::is_heap_until()`.

```
3.39.2.2  template<typename _RandomAccessIterator, typename _Compare> bool std::is_heap ( _RandomAccessIterator __first,
        _RandomAccessIterator __last, _Compare __comp ) [inline]
```

Determines whether a range is a heap using comparison functor.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__comp</code>	Comparison functor to use.

Returns

True if range is a heap, false otherwise.

Definition at line 532 of file `stl_heap.h`.

References `std::is_heap_until()`.

```
3.39.2.3  template<typename _RandomAccessIterator> _RandomAccessIterator std::is_heap_until ( _RandomAccessIterator
        __first, _RandomAccessIterator __last ) [inline]
```

Search the end of a heap.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.

Returns

An iterator pointing to the first element not in the heap.

This operation returns the last iterator *i* in [*__first*, *__last*) for which the range [*__first*, *i*) is a heap.

Definition at line 468 of file `stl_heap.h`.

References `std::distance()`.

3.39.2.4 `template<typename _RandomAccessIterator, typename _Compare> _RandomAccessIterator std::is_heap_until (`
`_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp) [inline]`

Search the end of a heap using comparison functor.

Parameters

<i>__first</i>	Start of range.
<i>__last</i>	End of range.
<i>__comp</i>	Comparison functor to use.

Returns

An iterator pointing to the first element not in the heap.

This operation returns the last iterator *i* in [*__first*, *__last*) for which the range [*__first*, *i*) is a heap. Comparisons are made using *__comp*.

Definition at line 496 of file `stl_heap.h`.

References `std::distance()`.

Referenced by `std::is_heap()`.

3.39.2.5 `template<typename _RandomAccessIterator> void std::make_heap (_RandomAccessIterator __first,`
`_RandomAccessIterator __last) [inline]`

Construct a heap over a range.

Parameters

<i>__first</i>	Start of heap.
<i>__last</i>	End of heap.

This operation makes the elements in [*__first*, *__last*) into a heap.

Definition at line 353 of file `stl_heap.h`.

3.39.2.6 `template<typename _RandomAccessIterator, typename _Compare> void std::make_heap (_RandomAccessIterator`
`__first, _RandomAccessIterator __last, _Compare __comp) [inline]`

Construct a heap over a range using comparison functor.

Parameters

<code>__first</code>	Start of heap.
<code>__last</code>	End of heap.
<code>__comp</code>	Comparison functor to use.

This operation makes the elements in `[__first,__last)` into a heap. Comparisons are made using `__comp`.

Definition at line 379 of file `stl_heap.h`.

Referenced by `std::priority_queue<_Tp, _Sequence, _Compare >::priority_queue()`.

3.39.2.7 `template<typename _RandomAccessIterator > void std::pop_heap (_RandomAccessIterator __first, _RandomAccessIterator __last) [inline]`

Pop an element off a heap.

Parameters

<code>__first</code>	Start of heap.
<code>__last</code>	End of heap.

Precondition

`[__first, __last)` is a valid, non-empty range.

This operation pops the top of the heap. The elements `__first` and `__last-1` are swapped and `[__first,__last-1)` is made into a heap.

Definition at line 265 of file `stl_heap.h`.

3.39.2.8 `template<typename _RandomAccessIterator, typename _Compare > void std::pop_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp) [inline]`

Pop an element off a heap using comparison functor.

Parameters

<code>__first</code>	Start of heap.
<code>__last</code>	End of heap.
<code>__comp</code>	Comparison functor to use.

This operation pops the top of the heap. The elements `__first` and `__last-1` are swapped and `[__first,__last-1)` is made into a heap. Comparisons are made using `comp`.

Definition at line 298 of file `stl_heap.h`.

Referenced by `std::priority_queue<_Tp, _Sequence, _Compare >::pop()`.

3.39.2.9 `template<typename _RandomAccessIterator > void std::push_heap (_RandomAccessIterator __first,
_RandomAccessIterator __last) [inline]`

Push an element onto a heap.

Parameters

<code>__first</code>	Start of heap.
<code>__last</code>	End of heap + element.

This operation pushes the element at `last-1` onto the valid heap over the range `[__first,__last-1)`. After completion, `[__first,__last)` is a valid heap.

Definition at line 150 of file `stl_heap.h`.

3.39.2.10 `template<typename _RandomAccessIterator , typename _Compare > void std::push_heap (_RandomAccessIterator
__first, _RandomAccessIterator __last, _Compare __comp) [inline]`

Push an element onto a heap using comparison functor.

Parameters

<code>__first</code>	Start of heap.
<code>__last</code>	End of heap + element.
<code>__comp</code>	Comparison functor.

This operation pushes the element at `__last-1` onto the valid heap over the range `[__first,__last-1)`. After completion, `[__first,__last)` is a valid heap. Compare operations are performed using `comp`.

Definition at line 185 of file `stl_heap.h`.

Referenced by `std::priority_queue<_Tp, _Sequence, _Compare >::push()`.

3.39.2.11 `template<typename _RandomAccessIterator > void std::sort_heap (_RandomAccessIterator __first,
_RandomAccessIterator __last) [inline]`

Sort a heap.

Parameters

<code>__first</code>	Start of heap.
<code>__last</code>	End of heap.

This operation sorts the valid heap in the range `[__first,__last)`.

Definition at line 414 of file `stl_heap.h`.

3.39.2.12 `template<typename _RandomAccessIterator, typename _Compare> void std::sort_heap (_RandomAccessIterator
__first, _RandomAccessIterator __last, _Compare __comp) [inline]`

Sort a heap using comparison functor.

Parameters

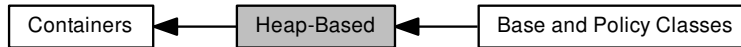
<code>__first</code>	Start of heap.
<code>__last</code>	End of heap.
<code>__comp</code>	Comparison functor to use.

This operation sorts the valid heap in the range [`__first`,`__last`). Comparisons are made using `__comp`.

Definition at line 441 of file `stl_heap.h`.

3.40 Heap-Based

Collaboration diagram for Heap-Based:



Modules

- [Base and Policy Classes](#)

Classes

- class [__gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc>](#)

Typedefs

- typedef `_Alloc __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc>::allocator_type`
- typedef `Cmp_Fn __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc>::cmp_fn`
- typedef `base_type::const_iterator __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc>::const_iterator`
- typedef `__rebind_va::const_pointer __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc>::const_pointer`
- typedef `__rebind_va::const_reference __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc>::const_reference`
- typedef `Tag __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc>::container_category`
- typedef `allocator_type::difference_type __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc>::difference_type`
- typedef `base_type::iterator __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc>::iterator`
- typedef `base_type::point_const_iterator __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc>::point_const_iterator`
- typedef `base_type::point_iterator __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc>::point_iterator`
- typedef `__rebind_va::pointer __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc>::pointer`
- typedef `__rebind_va::reference __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc>::reference`
- typedef `allocator_type::size_type __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc>::size_type`
- typedef `_Tv __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc>::value_type`

Functions

- `__gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc >::priority_queue` (const cmp_fn &r_cmp_fn)
- `template<typename It > __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc >::priority_queue` (It first_it, It last_it)
- `template<typename It > __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc >::priority_queue` (It first_it, It last_it, const cmp_fn &r_cmp_fn)
- `__gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc >::priority_queue` (const priority_queue &other)
- `priority_queue & __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc >::operator=` (const priority_queue &other)
- `void __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc >::swap` (priority_queue &other)

3.40.1 Detailed Description

3.40.2 Function Documentation

3.40.2.1 `template<typename _Tv, typename Cmp_Fn = std::less<_Tv>, typename Tag = pairing_heap_tag, typename _Alloc = std::allocator<char>> __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc >::priority_queue (const cmp_fn &r_cmp_fn) [inline]`

Constructor taking some policy objects. r_cmp_fn will be copied by the Cmp_Fn object of the container object.

Definition at line 116 of file priority_queue.hpp.

3.40.2.2 `template<typename _Tv, typename Cmp_Fn = std::less<_Tv>, typename Tag = pairing_heap_tag, typename _Alloc = std::allocator<char>> template<typename It > __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc >::priority_queue (It first_it, It last_it) [inline]`

Constructor taking __iterators to a range of value_types. The value_types between first_it and last_it will be inserted into the container object.

Definition at line 122 of file priority_queue.hpp.

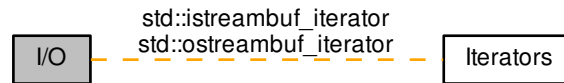
3.40.2.3 `template<typename _Tv, typename Cmp_Fn = std::less<_Tv>, typename Tag = pairing_heap_tag, typename _Alloc = std::allocator<char>> template<typename It > __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc >::priority_queue (It first_it, It last_it, const cmp_fn &r_cmp_fn) [inline]`

Constructor taking __iterators to a range of value_types and some policy objects The value_types between first_it and last_it will be inserted into the container object. r_cmp_fn will be copied by the cmp_fn object of the container object.

Definition at line 130 of file priority_queue.hpp.

3.41 I/O

Collaboration diagram for I/O:



Classes

- class [__gnu_cxx::stdio_filebuf< _CharT, _Traits >](#)
- class [__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >](#)
- class [std::basic_filebuf< _CharT, _Traits >](#)
- class [std::basic_fstream< _CharT, _Traits >](#)
- class [std::basic_ifstream< _CharT, _Traits >](#)
- class [std::basic_ios< _CharT, _Traits >](#)
- class [std::basic_iostream< _CharT, _Traits >](#)
- class [std::basic_istream< _CharT, _Traits >](#)
- class [std::basic_istreamstream< _CharT, _Traits, _Alloc >](#)
- class [std::basic_ofstream< _CharT, _Traits >](#)
- class [std::basic_ostream< _CharT, _Traits >](#)
- class [std::basic_ostreamstream< _CharT, _Traits, _Alloc >](#)
- class [std::basic_streambuf< _CharT, _Traits >](#)
- class [std::basic_stringbuf< _CharT, _Traits, _Alloc >](#)
- class [std::basic_stringstream< _CharT, _Traits, _Alloc >](#)
- class [std::ios_base](#)
- class [std::istreambuf_iterator< _CharT, _Traits >](#)
- class [std::ostreambuf_iterator< _CharT, _Traits >](#)

Typedefs

- typedef [basic_filebuf< char >](#) [std::filebuf](#)
- typedef [basic_fstream< char >](#) [std::fstream](#)
- typedef [basic_ifstream< char >](#) [std::ifstream](#)
- typedef [basic_ios< char >](#) [std::ios](#)
- typedef [basic_iostream< char >](#) [std::iostream](#)
- typedef [basic_istream< char >](#) [std::istream](#)
- typedef [basic_istreamstream< char >](#) [std::istreamstream](#)
- typedef [basic_ofstream< char >](#) [std::ofstream](#)
- typedef [basic_ostream< char >](#) [std::ostream](#)
- typedef [basic_ostreamstream< char >](#) [std::ostreamstream](#)
- typedef [basic_streambuf< char >](#) [std::streambuf](#)

- `typedef basic_stringbuf< char > std::stringbuf`
- `typedef basic_stringstream< char > std::stringstream`
- `typedef basic_filebuf< wchar_t > std::wfilebuf`
- `typedef basic_fstream< wchar_t > std::wfstream`
- `typedef basic_ifstream< wchar_t > std::wifstream`
- `typedef basic_ios< wchar_t > std::wios`
- `typedef basic_iostream< wchar_t > std::wiostream`
- `typedef basic_istream< wchar_t > std::wistream`
- `typedef basic_istreamstream< wchar_t > std::wistreamstream`
- `typedef basic_ofstream< wchar_t > std::wofstream`
- `typedef basic_ostream< wchar_t > std::wostream`
- `typedef basic_ostreamstream< wchar_t > std::wostringstream`
- `typedef basic_streambuf< wchar_t > std::wstreambuf`
- `typedef basic_stringbuf< wchar_t > std::wstringbuf`
- `typedef basic_stringstream< wchar_t > std::wstringstream`

3.41.1 Detailed Description

Nearly all of the I/O classes are parameterized on the type of characters they read and write. (The major exception is `ios_base` at the top of the hierarchy.) This is a change from pre-Standard streams, which were not templates.

For ease of use and compatibility, all of the `basic_*` I/O-related classes are given typedef names for both of the builtin character widths (wide and narrow). The typedefs are the same as the pre-Standard names, for example:

```
1 typedef basic_ifstream<char> ifstream;
```

Because properly forward-declaring these classes can be difficult, you should not do it yourself. Instead, include the `<iosfwd>` header, which contains only declarations of all the I/O classes as well as the typedefs. Trying to forward-declare the typedefs themselves (e.g., `class ostream;`) is not valid ISO C++.

For more specific declarations, see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/io.html#std.io.objects>

3.41.2 Typedef Documentation

3.41.2.1 `typedef basic_filebuf<char> std::filebuf`

Class for `char` file buffers.

Definition at line 159 of file `iosfwd`.

3.41.2.2 `typedef basic_fstream<char> std::fstream`

Class for `char` mixed input and output file streams.

Definition at line 168 of file `iosfwd`.

3.41.2.3 typedef basic_ifstream<char> std::ifstream

Class for `char` input file streams.

Definition at line 162 of file iosfwd.

3.41.2.4 typedef basic_ios<char> std::ios

Base class for `char` streams.

Definition at line 128 of file iosfwd.

3.41.2.5 typedef basic_iostream<char> std::iostream

Base class for `char` mixed input and output streams.

Definition at line 144 of file iosfwd.

3.41.2.6 typedef basic_istream<char> std::istream

Base class for `char` input streams.

Definition at line 138 of file iosfwd.

3.41.2.7 typedef basic_istream<char> std::istream

Class for `char` input memory streams.

Definition at line 150 of file iosfwd.

3.41.2.8 typedef basic_ofstream<char> std::ofstream

Class for `char` output file streams.

Definition at line 165 of file iosfwd.

3.41.2.9 typedef basic_ostream<char> std::ostream

Base class for `char` output streams.

Definition at line 141 of file iosfwd.

3.41.2.10 typedef basic_ostringstream<char> std::ostringstream

Class for `char` output memory streams.

Definition at line 153 of file iosfwd.

3.41.2.11 typedef basic_streambuf<char> std::streambuf

Base class for `char` buffers.

Definition at line 135 of file `iosfwd`.

3.41.2.12 typedef basic_stringbuf<char> std::stringbuf

Class for `char` memory buffers.

Definition at line 147 of file `iosfwd`.

3.41.2.13 typedef basic_stringstream<char> std::stringstream

Class for `char` mixed input and output memory streams.

Definition at line 156 of file `iosfwd`.

3.41.2.14 typedef basic_filebuf<wchar_t> std::wfilebuf

Class for `wchar_t` file buffers.

Definition at line 199 of file `iosfwd`.

3.41.2.15 typedef basic_fstream<wchar_t> std::wfstream

Class for `wchar_t` mixed input and output file streams.

Definition at line 208 of file `iosfwd`.

3.41.2.16 typedef basic_ifstream<wchar_t> std::wifstream

Class for `wchar_t` input file streams.

Definition at line 202 of file `iosfwd`.

3.41.2.17 typedef basic_ios<wchar_t> std::wios

Base class for `wchar_t` streams.

Definition at line 172 of file `iosfwd`.

3.41.2.18 typedef basic_iostream<wchar_t> std::wiostream

Base class for `wchar_t` mixed input and output streams.

Definition at line 184 of file `iosfwd`.

3.41.2.19 `typedef basic_istream<wchar_t> std::wistream`

Base class for `wchar_t` input streams.

Definition at line 178 of file `iosfwd`.

3.41.2.20 `typedef basic_istream<wchar_t> std::wistream`

Class for `wchar_t` input memory streams.

Definition at line 190 of file `iosfwd`.

3.41.2.21 `typedef basic_ofstream<wchar_t> std::wofstream`

Class for `wchar_t` output file streams.

Definition at line 205 of file `iosfwd`.

3.41.2.22 `typedef basic_ostream<wchar_t> std::wostream`

Base class for `wchar_t` output streams.

Definition at line 181 of file `iosfwd`.

3.41.2.23 `typedef basic_ostream<wchar_t> std::wostream`

Class for `wchar_t` output memory streams.

Definition at line 193 of file `iosfwd`.

3.41.2.24 `typedef basic_streambuf<wchar_t> std::wstreambuf`

Base class for `wchar_t` buffers.

Definition at line 175 of file `iosfwd`.

3.41.2.25 `typedef basic_stringbuf<wchar_t> std::wstringbuf`

Class for `wchar_t` memory buffers.

Definition at line 187 of file `iosfwd`.

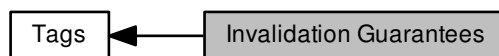
3.41.2.26 `typedef basic_stringstream<wchar_t> std::wstringstream`

Class for `wchar_t` mixed input and output memory streams.

Definition at line 196 of file `iosfwd`.

3.42 Invalidation Guarantees

Collaboration diagram for Invalidation Guarantees:



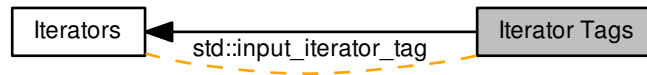
Classes

- struct [__gnu_pbds::basic_invalidation_guarantee](#)
- struct [__gnu_pbds::point_invalidation_guarantee](#)
- struct [__gnu_pbds::range_invalidation_guarantee](#)

3.42.1 Detailed Description

3.43 Iterator Tags

Collaboration diagram for Iterator Tags:



Classes

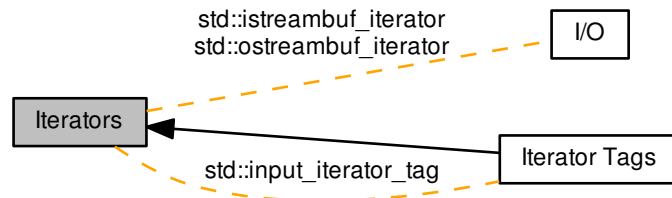
- struct [std::bidirectional_iterator_tag](#)
- struct [std::forward_iterator_tag](#)
- struct [std::input_iterator_tag](#)
- struct [std::output_iterator_tag](#)
- struct [std::random_access_iterator_tag](#)

3.43.1 Detailed Description

These are empty types, used to distinguish different iterators. The distinction is not made by what they contain, but simply by what they are. Different underlying algorithms can then be used based on the different operations supported by different iterator types.

3.44 Iterators

Collaboration diagram for Iterators:



Modules

- [Iterator Tags](#)

Classes

- struct [std::__iterator_traits<_Iterator, typename>](#)
- class [std::back_insert_iterator<_Container>](#)
- class [std::front_insert_iterator<_Container>](#)
- struct [std::input_iterator_tag](#)
- class [std::insert_iterator<_Container>](#)
- class [std::istream_iterator<_Tp, _CharT, _Traits, _Dist>](#)
- class [std::istreambuf_iterator<_CharT, _Traits>](#)
- struct [std::iterator<_Category, _Tp, _Distance, _Pointer, _Reference>](#)
- struct [std::iterator_traits<_Tp*>](#)
- struct [std::iterator_traits<const _Tp*>](#)
- class [std::move_iterator<_Iterator>](#)
- class [std::ostream_iterator<_Tp, _CharT, _Traits>](#)
- class [std::ostreambuf_iterator<_CharT, _Traits>](#)
- class [std::reverse_iterator<_Iterator>](#)

Macros

- `#define __cpp_lib_make_reverse_iterator`

Functions

- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, ostreambuf_iterator< _CharT > >::__type std::__↵`
`copy_move_a2 (_CharT * __first, _CharT * __last, ostreambuf_iterator< _CharT > __result)`
- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, ostreambuf_iterator< _CharT > >::__type std::__↵`
`copy_move_a2 (const _CharT * __first, const _CharT * __last, ostreambuf_iterator< _CharT > __result)`
- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, _CharT * >::__type std:: copy_move_a2`
`(istreambuf_iterator< _CharT > __first, istreambuf_iterator< _CharT > __last, _CharT * __result)`
- `template<typename _Iter >`
`iterator_traits< _Iter >::iterator_category std:: iterator_category (const _Iter &)`
- `template<typename _Iterator , typename _ReturnType = typename conditional< __move_if_noexcept_cond <typename iterator_traits<_↵`
`Iterator>::value_type>::__value, _Iterator, move_iterator< _Iterator>>::__type>`
`_ReturnType std::__make_move_if_noexcept_iterator (_Iterator __i)`
- `template<typename _Tp , typename _ReturnType = typename conditional< __move_if_noexcept_cond<_Tp>::__value, const _Tp*, move↵`
`_iterator< _Tp*>>::__type>`
`_ReturnType std::__make_move_if_noexcept_iterator (_Tp * __i)`
- `template<typename _Iterator >`
`reverse_iterator< _Iterator > std::__make_reverse_iterator (_Iterator __i)`
- `template<typename _Iterator >`
`auto std::__miter_base (reverse_iterator< _Iterator > __it) -> decltype(__make_reverse_iterator(__miter_↵`
`base(__it.base()))`
- `template<typename _Iterator >`
`auto std::__niter_base (reverse_iterator< _Iterator > __it) -> decltype(__make_reverse_iterator(__niter_↵`
`base(__it.base()))`
- `template<typename _Container >`
`back_insert_iterator< _Container > std::back_inserter (_Container & __x)`
- `template<typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, ostreambuf_iterator< _CharT > >::__type std::copy`
`(istreambuf_iterator< _CharT > __first, istreambuf_iterator< _CharT > __last, ostreambuf_iterator< _CharT >`
`__result)`
- `template<typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, istreambuf_iterator< _CharT > >::__type std::find`
`(istreambuf_iterator< _CharT > __first, istreambuf_iterator< _CharT > __last, const _CharT & __val)`
- `template<typename _Container >`
`front_insert_iterator< _Container > std::front_inserter (_Container & __x)`
- `template<typename _Container , typename _Iterator >`
`insert_iterator< _Container > std::inserter (_Container & __x, _Iterator __i)`
- `template<typename _Iterator >`
`move_iterator< _Iterator > std::make_move_iterator (_Iterator __i)`
- `template<typename _Iterator >`
`reverse_iterator< _Iterator > std::make_reverse_iterator (_Iterator __i)`
- `template<class _Tp , class _CharT , class _Traits , class _Dist >`
`bool std::operator!= (const istream_iterator< _Tp, _CharT, _Traits, _Dist > & __x, const istream_iterator< _Tp,`
`_CharT, _Traits, _Dist > & __y)`
- `template<typename _CharT , typename _Traits >`
`bool std::operator!= (const istreambuf_iterator< _CharT, _Traits > & __a, const istreambuf_iterator< _CharT,`
`_Traits > & __b)`
- `template<typename _Iterator >`
`bool std::operator!= (const reverse_iterator< _Iterator > & __x, const reverse_iterator< _Iterator > & __y)`

- `template<typename _IteratorL, typename _IteratorR >`
`bool std::operator!= (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool std::operator!= (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _Iterator >`
`bool std::operator!= (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _Iterator >`
`reverse_iterator< _Iterator > std::operator+ (typename reverse_iterator< _Iterator >::difference_type __n, const reverse_iterator< _Iterator > &__x)`
- `template<typename _Iterator >`
`move_iterator< _Iterator > std::operator+ (typename move_iterator< _Iterator >::difference_type __n, const move_iterator< _Iterator > &__x)`
- `template<typename _Iterator >`
`auto std::operator- (const reverse_iterator< _Iterator > &__x, const reverse_iterator< _Iterator > &__y) ->`
`decltype(__x.base()-__y.base())`
- `template<typename _IteratorL, typename _IteratorR >`
`auto std::operator- (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y) ->`
`decltype(__y.base()-__x.base())`
- `template<typename _IteratorL, typename _IteratorR >`
`auto std::operator- (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y) ->`
`decltype(__x.base()-__y.base())`
- `template<typename _Iterator >`
`auto std::operator- (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y) ->`
`decltype(__x.base()-__y.base())`
- `template<typename _Iterator >`
`bool std::operator< (const reverse_iterator< _Iterator > &__x, const reverse_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool std::operator< (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool std::operator< (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _Iterator >`
`bool std::operator< (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _Iterator >`
`bool std::operator<= (const reverse_iterator< _Iterator > &__x, const reverse_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool std::operator<= (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool std::operator<= (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _Iterator >`
`bool std::operator<= (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _Tp, typename _CharT, typename _Traits, typename _Dist >`
`bool std::operator== (const istream_iterator< _Tp, _CharT, _Traits, _Dist > &__x, const istream_iterator< _Tp, _CharT, _Traits, _Dist > &__y)`
- `template<typename _CharT, typename _Traits >`
`bool std::operator== (const istreambuf_iterator< _CharT, _Traits > &__a, const istreambuf_iterator< _CharT, _Traits > &__b)`
- `template<typename _Iterator >`
`bool std::operator== (const reverse_iterator< _Iterator > &__x, const reverse_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool std::operator== (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool std::operator== (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`

- `template<typename _Iterator >`
`bool std::operator== (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _Iterator >`
`bool std::operator> (const reverse_iterator< _Iterator > &__x, const reverse_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool std::operator> (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool std::operator> (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _Iterator >`
`bool std::operator> (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _Iterator >`
`bool std::operator>= (const reverse_iterator< _Iterator > &__x, const reverse_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool std::operator>= (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool std::operator>= (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _Iterator >`
`bool std::operator>= (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`

3.44.1 Detailed Description

Abstractions for uniform iterating through various underlying types.

3.44.2 Function Documentation

3.44.2.1 `template<typename _Iter > iterator_traits<_Iter>::iterator_category std::__iterator_category (const _Iter &)`
`[inline]`

This function is not a part of the C++ standard but is syntactic sugar for internal library use only.

Definition at line 204 of file `stl_iterator_base_types.h`.

Referenced by `__gnu_debug::__check_string()`, `std::__find_if()`, `std::__find_if_not()`, `__gnu_debug::__get_distance()`, `std::__search_n_aux()`, `std::advance()`, `std::copy_n()`, `std::distance()`, `std::find_end()`, `std::includes()`, `std::next__↵` `permutation()`, `std::partition()`, `std::reverse()`, `std::_V2::rotate()`, `std::uninitialized_copy_n()`, and `std::unique_copy()`.

3.44.2.2 `template<typename _Container > back_insert_iterator<_Container> std::back_inserter (_Container & __x)`
`[inline]`

Parameters

<code>__↵</code>	A container of arbitrary type.
<code>__x</code>	

Returns

An instance of `back_insert_iterator` working on `__x`.

This wrapper function helps in creating `back_insert_iterator` instances. Typing the name of the iterator requires knowing the precise full type of the container, which can be tedious and impedes generic programming. Using this function lets you take advantage of automatic template parameter deduction, making the compiler match the correct types for you.

Definition at line 527 of file `bits/stl_iterator.h`.

Referenced by `std::match_results<_Bi_iter>::format()`, `std::operator>>()`, and `std::regex_replace()`.

3.44.2.3 `template<typename _Container> front_insert_iterator<_Container> std::front_inserter (_Container & __x)`
`[inline]`

Parameters

<code>__x</code>	A container of arbitrary type.
------------------	--------------------------------

Returns

An instance of `front_insert_iterator` working on `x`.

This wrapper function helps in creating `front_insert_iterator` instances. Typing the name of the iterator requires knowing the precise full type of the container, which can be tedious and impedes generic programming. Using this function lets you take advantage of automatic template parameter deduction, making the compiler match the correct types for you.

Definition at line 618 of file `bits/stl_iterator.h`.

3.44.2.4 `template<typename _Container, typename _Iterator> insert_iterator<_Container> std::inserter (_Container & __x,`
`_Iterator __i) [inline]`

Parameters

<code>__x</code>	A container of arbitrary type.
------------------	--------------------------------

Returns

An instance of `insert_iterator` working on `__x`.

This wrapper function helps in creating `insert_iterator` instances. Typing the name of the iterator requires knowing the precise full type of the container, which can be tedious and impedes generic programming. Using this function lets you take advantage of automatic template parameter deduction, making the compiler match the correct types for you.

Definition at line 732 of file `bits/stl_iterator.h`.

References `std::reverse_iterator<_Iterator>::base()`, `std::reverse_iterator<_Iterator>::operator*()`, `std::reverse_iterator<_Iterator>::operator+()`, `std::reverse_iterator<_Iterator>::operator++()`, `std::reverse_iterator<_Iterator`

`>::operator+=(())`, `std::reverse_iterator< _Iterator >::operator-()`, `std::reverse_iterator< _Iterator >::operator--()`, `std::reverse_iterator< _Iterator >::operator-=()`, `std::reverse_iterator< _Iterator >::operator->()`, and `std::reverse_iterator< _Iterator >::operator[]()`.

Referenced by `std::deque< _Tp, _Alloc >::_M_pop_front_aux()`.

3.44.2.5 `template<typename _Iterator> reverse_iterator<_Iterator> std::make_reverse_iterator (_Iterator __i) [inline]`

Generator function for `reverse_iterator`.

Definition at line 413 of file `bits/stl_iterator.h`.

References `std::reverse_iterator< _Iterator >::base()`.

3.44.2.6 `template<class _Tp, class _CharT, class _Traits, class _Dist> bool std::operator!= (const istream_iterator< _Tp, _CharT, _Traits, _Dist> &__x, const istream_iterator< _Tp, _CharT, _Traits, _Dist> &__y) [inline]`

Return false if `x` and `y` are both end or not end, or `x` and `y` are the same.

Definition at line 137 of file `stream_iterator.h`.

3.44.2.7 `template<typename _Tp, typename _CharT, typename _Traits, typename _Dist> bool std::operator== (const istream_iterator< _Tp, _CharT, _Traits, _Dist> &__x, const istream_iterator< _Tp, _CharT, _Traits, _Dist> &__y) [inline]`

Return true if `x` and `y` are both end or not end, or `x` and `y` are the same.

Definition at line 130 of file `stream_iterator.h`.

3.44.2.8 `template<typename _Iterator> bool std::operator== (const reverse_iterator< _Iterator> &__x, const reverse_iterator< _Iterator> &__y) [inline]`

Parameters

<code>__x</code>	A <code>reverse_iterator</code> .
<code>__y</code>	A <code>reverse_iterator</code> .

Returns

A simple `bool`.

Reverse iterators forward many operations to their underlying `base()` iterators. Others are implemented in terms of one another.

Definition at line 292 of file `bits/stl_iterator.h`.

References `std::reverse_iterator< _Iterator >::base()`, `std::reverse_iterator< _Iterator >::operator+()`, and `std::reverse_iterator< _Iterator >::operator-()`.

3.45 List-Based

Collaboration diagram for List-Based:



Classes

- class `__gnu_pbds::list_update< Key, Mapped, Eq_Fn, Update_Policy, _Alloc >`

Macros

- `#define PB_DS_LU_BASE`

3.45.1 Detailed Description

3.46 Locales

Classes

- class `std::codecvt< _InternT, _ExternT, _StateT >`
- class `std::ctype< _CharT >`
- class `std::ctype< char >`
- class `std::ctype< wchar_t >`
- class `std::locale`
- class `std::locale::facet`
- class `std::locale::id`
- class `std::messages< _CharT >`
- struct `std::messages_base`
- class `std::money_base`
- class `std::money_get< _CharT, _InIter >`
- class `std::money_put< _CharT, _OutIter >`
- class `std::moneypunct< _CharT, _Intl >`
- class `std::num_get< _CharT, _InIter >`
- class `std::num_put< _CharT, _OutIter >`
- class `std::numpunct< _CharT >`
- class `std::time_base`
- class `std::time_get< _CharT, _InIter >`
- class `std::time_put< _CharT, _OutIter >`
- class `std::wbuffer_convert< _Codecvt, _Elem, _Tr >`
- class `std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >`

Functions

- `template<typename _OutStr, typename _InChar, typename _Codecvt, typename _State, typename _Fn >`
`bool std::__do_str_codecvt (const _InChar *__first, const _InChar *__last, _OutStr &__outstr, const _Codecvt &__cvt, _State &__state, size_t &__count, _Fn __fn)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool std::__str_codecvt_in (const char *__first, const char *__last, basic_string< _CharT, _Traits, _Alloc > &__outstr, const codecvt< _CharT, char, _State > &__cvt, _State &__state, size_t &__count)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool std::__str_codecvt_in (const char *__first, const char *__last, basic_string< _CharT, _Traits, _Alloc > &__outstr, const codecvt< _CharT, char, _State > &__cvt)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool std::__str_codecvt_out (const _CharT *__first, const _CharT *__last, basic_string< char, _Traits, _Alloc > &__outstr, const codecvt< _CharT, char, _State > &__cvt, _State &__state, size_t &__count)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool std::__str_codecvt_out (const _CharT *__first, const _CharT *__last, basic_string< char, _Traits, _Alloc > &__outstr, const codecvt< _CharT, char, _State > &__cvt)`
- `template<typename _Facet >`
`bool std::has_facet (const locale &__loc) throw ()`
- `template<typename _Facet >`
`const _Facet & std::use_facet (const locale &__loc)`

3.46.1 Detailed Description

Classes and functions for internationalization and localization.

3.46.2 Function Documentation

3.46.2.1 `template<typename _Facet> bool std::has_facet (const locale & __loc) throw`

Test for the presence of a facet.

`has_facet` tests the locale argument for the presence of the facet type provided as the template parameter. Facets derived from the facet parameter will also return true.

Template Parameters

<code>_Facet</code>	The facet type to test the presence of.
---------------------	---

Parameters

<code>__loc</code>	The locale to test.
--------------------	---------------------

Returns

true if `__loc` contains a facet of type `_Facet`, else false.

Definition at line 104 of file `locale_classes.tcc`.

3.46.2.2 `template<typename _Facet> const _Facet & std::use_facet (const locale & __loc)`

Return a facet.

`use_facet` looks for and returns a reference to a facet of type `Facet` where `Facet` is the template parameter. If `has_facet(locale)` is true, there is a suitable facet to return. It throws `std::bad_cast` if the locale doesn't contain a facet of type `Facet`.

Template Parameters

<code>_Facet</code>	The facet type to access.
---------------------	---------------------------

Parameters

<code>__loc</code>	The locale to use.
--------------------	--------------------

Returns

Reference to facet of type `Facet`.

Exceptions

<code>std::bad_cast</code>	if <code>__loc</code> doesn't contain a facet of type <code>_Facet</code> .
----------------------------	---

Definition at line 132 of file locale_classes.tcc.

References `std::collate<_CharT>::do_compare()`.

3.47 Mathematical Special Functions

Collaboration diagram for Mathematical Special Functions:



Functions

- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::assoc_laguerre` (unsigned int __n, unsigned int __m, _Tp __x)
- `float std::assoc_laguerref` (unsigned int __n, unsigned int __m, float __x)
- `long double std::assoc_laguerrel` (unsigned int __n, unsigned int __m, long double __x)
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::assoc_legendre` (unsigned int __l, unsigned int __m, _Tp __x)
- `float std::assoc_legendref` (unsigned int __l, unsigned int __m, float __x)
- `long double std::assoc_legendrel` (unsigned int __l, unsigned int __m, long double __x)
- `template<typename _Tpa, typename _Tpb >`
`__gnu_cxx::__promote_2< _Tpa, _Tpb >::__type std::beta` (_Tpa __a, _Tpb __b)
- `float std::betaf` (float __a, float __b)
- `long double std::betal` (long double __a, long double __b)
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::comp_ellint_1` (_Tp __k)
- `float std::comp_ellint_1f` (float __k)
- `long double std::comp_ellint_1l` (long double __k)
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::comp_ellint_2` (_Tp __k)
- `float std::comp_ellint_2f` (float __k)
- `long double std::comp_ellint_2l` (long double __k)
- `template<typename _Tp, typename _Tpn >`
`__gnu_cxx::__promote_2< _Tp, _Tpn >::__type std::comp_ellint_3` (_Tp __k, _Tpn __nu)
- `float std::comp_ellint_3f` (float __k, float __nu)
- `long double std::comp_ellint_3l` (long double __k, long double __nu)
- `template<typename _Tpnu, typename _Tp >`
`__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::cyl_bessel_i` (_Tpnu __nu, _Tp __x)
- `float std::cyl_bessel_if` (float __nu, float __x)
- `long double std::cyl_bessel_il` (long double __nu, long double __x)
- `template<typename _Tpnu, typename _Tp >`
`__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::cyl_bessel_j` (_Tpnu __nu, _Tp __x)
- `float std::cyl_bessel_jf` (float __nu, float __x)
- `long double std::cyl_bessel_jl` (long double __nu, long double __x)
- `template<typename _Tpnu, typename _Tp >`
`__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::cyl_bessel_k` (_Tpnu __nu, _Tp __x)

- float [std::cyl_bessel_kf](#) (float __nu, float __x)
- long double [std::cyl_bessel_kl](#) (long double __nu, long double __x)
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2<_Tpnu, _Tp >::__type [std::cyl_neumann](#) (_Tpnu __nu, _Tp __x)
- float [std::cyl_neumannf](#) (float __nu, float __x)
- long double [std::cyl_neumannl](#) (long double __nu, long double __x)
- template<typename _Tp, typename _Tpp >
__gnu_cxx::__promote_2<_Tp, _Tpp >::__type [std::ellint_1](#) (_Tp __k, _Tpp __phi)
- float [std::ellint_1f](#) (float __k, float __phi)
- long double [std::ellint_1l](#) (long double __k, long double __phi)
- template<typename _Tp, typename _Tpp >
__gnu_cxx::__promote_2<_Tp, _Tpp >::__type [std::ellint_2](#) (_Tp __k, _Tpp __phi)
- float [std::ellint_2f](#) (float __k, float __phi)
- long double [std::ellint_2l](#) (long double __k, long double __phi)
- template<typename _Tp, typename _Tpn, typename _Tpp >
__gnu_cxx::__promote_3<_Tp, _Tpn, _Tpp >::__type [std::ellint_3](#) (_Tp __k, _Tpn __nu, _Tpp __phi)
- float [std::ellint_3f](#) (float __k, float __nu, float __phi)
- long double [std::ellint_3l](#) (long double __k, long double __nu, long double __phi)
- template<typename _Tp >
__gnu_cxx::__promote<_Tp >::__type [std::expint](#) (_Tp __x)
- float [std::expintf](#) (float __x)
- long double [std::expintl](#) (long double __x)
- template<typename _Tp >
__gnu_cxx::__promote<_Tp >::__type [std::hermite](#) (unsigned int __n, _Tp __x)
- float [std::hermitef](#) (unsigned int __n, float __x)
- long double [std::hermitel](#) (unsigned int __n, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote<_Tp >::__type [std::laguerre](#) (unsigned int __n, _Tp __x)
- float [std::laguerref](#) (unsigned int __n, float __x)
- long double [std::laguerrel](#) (unsigned int __n, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote<_Tp >::__type [std::legendre](#) (unsigned int __l, _Tp __x)
- float [std::legendref](#) (unsigned int __l, float __x)
- long double [std::legendrel](#) (unsigned int __l, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote<_Tp >::__type [std::riemann_zeta](#) (_Tp __s)
- float [std::riemann_zetaf](#) (float __s)
- long double [std::riemann_zetal](#) (long double __s)
- template<typename _Tp >
__gnu_cxx::__promote<_Tp >::__type [std::sph_bessel](#) (unsigned int __n, _Tp __x)
- float [std::sph_besself](#) (unsigned int __n, float __x)
- long double [std::sph_bessell](#) (unsigned int __n, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote<_Tp >::__type [std::sph_legendre](#) (unsigned int __l, unsigned int __m, _Tp __theta)
- float [std::sph_legendref](#) (unsigned int __l, unsigned int __m, float __theta)
- long double [std::sph_legendrel](#) (unsigned int __l, unsigned int __m, long double __theta)
- template<typename _Tp >
__gnu_cxx::__promote<_Tp >::__type [std::sph_neumann](#) (unsigned int __n, _Tp __x)
- float [std::sph_neumannf](#) (unsigned int __n, float __x)
- long double [std::sph_neumannl](#) (unsigned int __n, long double __x)

3.47.1 Detailed Description

A collection of advanced mathematical special functions, defined by ISO/IEC IS 29124.

3.47.2 Function Documentation

3.47.2.1 `template<typename _Tp> __gnu_cxx::__promote<_Tp>::__type std::assoc_laguerre (unsigned int __n, unsigned int __m, _Tp __x) [inline]`

Return the associated Laguerre polynomial of nonnegative order n , nonnegative degree m and real argument $x \leftarrow$: $L_n^m(x)$.

The associated Laguerre function of real degree α , $L_n^\alpha(x)$, is defined by

$$L_n^\alpha(x) = \frac{(\alpha+1)_n}{n!} {}_1F_1(-n; \alpha+1; x)$$

where $(\alpha)_n$ is the Pochhammer symbol and ${}_1F_1(a; c; x)$ is the confluent hypergeometric function.

The associated Laguerre polynomial is defined for integral degree $\alpha = m$ by:

$$L_n^m(x) = (-1)^m \frac{d^m}{dx^m} L_{n+m}(x)$$

where the Laguerre polynomial is defined by:

$$L_n(x) = \frac{e^x}{n!} \frac{d^n}{dx^n} (x^n e^{-x})$$

and $x \geq 0$.

See also

`laguerre` for details of the Laguerre function of degree n

Template Parameters

<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .
------------------	--

Parameters

<code>__n</code>	The order of the Laguerre function, <code>__n</code> ≥ 0 .
<code>__m</code>	The degree of the Laguerre function, <code>__m</code> ≥ 0 .
<code>__x</code>	The argument of the Laguerre function, <code>__x</code> ≥ 0 .

Exceptions

<code>std::domain_error</code>	if <code>__x < 0</code> .
--------------------------------	------------------------------

Definition at line 252 of file `specfun.h`.

3.47.2.2 `float std::assoc_laguerref (unsigned int __n, unsigned int __m, float __x)` `[inline]`

Return the associated Laguerre polynomial of order `n`, degree `m`: $L_n^m(x)$ for `float` argument.

See also

`assoc_laguerre` for more details.

Definition at line 206 of file `specfun.h`.

3.47.2.3 `long double std::assoc_laguerrel (unsigned int __n, unsigned int __m, long double __x)` `[inline]`

Return the associated Laguerre polynomial of order `n`, degree `m`: $L_n^m(x)$.

See also

`assoc_laguerre` for more details.

Definition at line 216 of file `specfun.h`.

3.47.2.4 `template<typename _Tp> __gnu_cxx::__promote<_Tp>::__type std::assoc_legendre (unsigned int __l, unsigned int __m, _Tp __x)` `[inline]`

Return the associated Legendre function of degree `l` and order `m`.

The associated Legendre function is derived from the Legendre function $P_l(x)$ by the Rodrigues formula:

$$P_l^m(x) = (1 - x^2)^{m/2} \frac{d^m}{dx^m} P_l(x)$$

See also

`legendre` for details of the Legendre function of degree `l`

Template Parameters

<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .
------------------	--

Parameters

<code>__l</code>	The degree <code>__l >= 0</code> .
<code>__m</code>	The order <code>__m <= 1</code> .
<code>__x</code>	The argument, <code>abs (__x) <= 1</code> .

Exceptions

<code>std::domain_error</code>	if <code>abs (__x) > 1</code> .
--------------------------------	------------------------------------

Definition at line 298 of file `specfun.h`.

3.47.2.5 `float std::assoc_legendref (unsigned int __l, unsigned int __m, float __x) [inline]`

Return the associated Legendre function of degree `l` and order `m` for `float` argument.

See also

`assoc_legendre` for more details.

Definition at line 267 of file `specfun.h`.

3.47.2.6 `long double std::assoc_legendrel (unsigned int __l, unsigned int __m, long double __x) [inline]`

Return the associated Legendre function of degree `l` and order `m`.

See also

`assoc_legendre` for more details.

Definition at line 276 of file `specfun.h`.

3.47.2.7 `template<typename _Tpa, typename _Tpb> __gnu_cxx::__promote_2<_Tpa, _Tpb>::__type std::beta (_Tpa __a, _Tpb __b) [inline]`

Return the beta function, $B(a, b)$, for real parameters `a`, `b`.

The beta function is defined by

$$B(a, b) = \int_0^1 t^{a-1} (1-t)^{b-1} dt = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$$

where $a > 0$ and $b > 0$

Template Parameters

<code>_Tpa</code>	The floating-point type of the parameter <code>__a</code> .
<code>_Tpb</code>	The floating-point type of the parameter <code>__b</code> .

Parameters

<code>__a</code>	The first argument of the beta function, <code>__a > 0</code> .
<code>__b</code>	The second argument of the beta function, <code>__b > 0</code> .

Exceptions

<code>std::domain_error</code>	if <code>__a < 0</code> or <code>__b < 0</code> .
--------------------------------	---

Definition at line 343 of file `specfun.h`.

3.47.2.8 `float std::betaf (float __a, float __b) [inline]`

Return the beta function, $B(a, b)$, for `float` parameters `a`, `b`.

See also

`beta` for more details.

Definition at line 312 of file `specfun.h`.

3.47.2.9 `long double std::betal (long double __a, long double __b) [inline]`

Return the beta function, $B(a, b)$, for long double parameters `a`, `b`.

See also

`beta` for more details.

Definition at line 322 of file `specfun.h`.

3.47.2.10 `template<typename _Tp> __gnu_cxx::__promote<_Tp>::__type std::comp_ellint_1 (_Tp __k) [inline]`

Return the complete elliptic integral of the first kind $K(k)$ for real modulus `k`.

The complete elliptic integral of the first kind is defined as

$$K(k) = F(k, \pi/2) = \int_0^{\pi/2} \frac{d\theta}{\sqrt{1 - k^2 \sin^2 \theta}}$$

where $F(k, \phi)$ is the incomplete elliptic integral of the first kind and the modulus $|k| \leq 1$.

See also

`ellint_1` for details of the incomplete elliptic function of the first kind.

Template Parameters

<code>_Tp</code>	The floating-point type of the modulus <code>__k</code> .
------------------	---

Parameters

<code>__k</code>	The modulus, <code>abs (__k) <= 1</code>
------------------	---

Exceptions

<code>std::domain_error</code>	if <code>abs (__k) > 1</code> .
--------------------------------	------------------------------------

Definition at line 391 of file `specfun.h`.

3.47.2.11 `float std::comp_ellint_1f (float __k) [inline]`

Return the complete elliptic integral of the first kind $E(k)$ for `float` modulus `k`.

See also

`comp_ellint_1` for details.

Definition at line 358 of file `specfun.h`.

3.47.2.12 `long double std::comp_ellint_1l (long double __k) [inline]`

Return the complete elliptic integral of the first kind $E(k)$ for long double modulus `k`.

See also

`comp_ellint_1` for details.

Definition at line 368 of file `specfun.h`.

3.47.2.13 `template<typename _Tp> __gnu_cxx::__promote<_Tp>::__type std::comp_ellint_2 (_Tp __k) [inline]`

Return the complete elliptic integral of the second kind $E(k)$ for real modulus `k`.

The complete elliptic integral of the second kind is defined as

$$E(k) = E(k, \pi/2) = \int_0^{\pi/2} \sqrt{1 - k^2 \sin^2 \theta} d\theta$$

where $E(k, \phi)$ is the incomplete elliptic integral of the second kind and the modulus $|k| \leq 1$.

See also

`ellint_2` for details of the incomplete elliptic function of the second kind.

Template Parameters

<code>_Tp</code>	The floating-point type of the modulus <code>__k</code> .
------------------	---

Parameters

<code>__k</code>	The modulus, <code>abs (__k) <= 1</code>
------------------	---

Exceptions

<code>std::domain_error</code>	if <code>abs (__k) > 1</code> .
--------------------------------	------------------------------------

Definition at line 438 of file `specfun.h`.

3.47.2.14 `float std::comp_ellint_2f (float __k) [inline]`

Return the complete elliptic integral of the second kind $E(k)$ for `float` modulus `k`.

See also

`comp_ellint_2` for details.

Definition at line 406 of file `specfun.h`.

3.47.2.15 `long double std::comp_ellint_2l (long double __k) [inline]`

Return the complete elliptic integral of the second kind $E(k)$ for long double modulus `k`.

See also

`comp_ellint_2` for details.

Definition at line 416 of file `specfun.h`.

3.47.2.16 `template<typename _Tp, typename _Tpn> __gnu_cxx::__promote_2<_Tp, _Tpn>::__type std::comp_ellint_3 (_Tp __k, _Tpn __nu) [inline]`

Return the complete elliptic integral of the third kind $\Pi(k, \nu) = \Pi(k, \nu, \pi/2)$ for real modulus `k`.

The complete elliptic integral of the third kind is defined as

$$\Pi(k, \nu) = \Pi(k, \nu, \pi/2) = \int_0^{\pi/2} \frac{d\theta}{(1 - \nu \sin^2 \theta) \sqrt{1 - k^2 \sin^2 \theta}}$$

where $\Pi(k, \nu, \phi)$ is the incomplete elliptic integral of the second kind and the modulus $|k| \leq 1$.

See also

`ellint_3` for details of the incomplete elliptic function of the third kind.

Template Parameters

<code>_Tp</code>	The floating-point type of the modulus <code>__k</code> .
<code>_Tpn</code>	The floating-point type of the argument <code>__nu</code> .

Parameters

<code>__k</code>	The modulus, <code>abs (__k) <= 1</code>
<code>__nu</code>	The argument

Exceptions

<code>std::domain_error</code>	if <code>abs (__k) > 1</code> .
--------------------------------	------------------------------------

Definition at line 489 of file `specfun.h`.

3.47.2.17 `float std::comp_ellint_3f (float __k, float __nu) [inline]`

Return the complete elliptic integral of the third kind $\Pi(k, \nu)$ for `float` modulus `k`.

See also

`comp_ellint_3` for details.

Definition at line 453 of file `specfun.h`.

3.47.2.18 `long double std::comp_ellint_3l (long double __k, long double __nu) [inline]`

Return the complete elliptic integral of the third kind $\Pi(k, \nu)$ for `long double` modulus `k`.

See also

`comp_ellint_3` for details.

Definition at line 463 of file `specfun.h`.

3.47.2.19 `template<typename _Tpnu, typename _Tp> __gnu_cxx::__promote_2<_Tpnu, _Tp>::__type std::cyl_bessel_i (_Tpnu __nu, _Tp __x) [inline]`

Return the regular modified Bessel function $I_\nu(x)$ for real order ν and argument $x \geq 0$.

The regular modified cylindrical Bessel function is:

$$I_\nu(x) = i^{-\nu} J_\nu(ix) = \sum_{k=0}^{\infty} \frac{(x/2)^{\nu+2k}}{k! \Gamma(\nu + k + 1)}$$

Template Parameters

<code>_Tpnu</code>	The floating-point type of the order <code>__nu</code> .
<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .

Parameters

<code>__nu</code>	The order
<code>__x</code>	The argument, <code>__x >= 0</code>

Exceptions

<code>std::domain_error</code>	if <code>__x < 0</code> .
--------------------------------	------------------------------

Definition at line 535 of file `specfun.h`.

3.47.2.20 `float std::cyl_bessel_if (float __nu, float __x) [inline]`

Return the regular modified Bessel function $I_\nu(x)$ for `float` order ν and argument $x \geq 0$.

See also

`cyl_bessel_i` for setails.

Definition at line 504 of file `specfun.h`.

3.47.2.21 `long double std::cyl_bessel_il (long double __nu, long double __x) [inline]`

Return the regular modified Bessel function $I_\nu(x)$ for `long double` order ν and argument $x \geq 0$.

See also

`cyl_bessel_i` for setails.

Definition at line 514 of file `specfun.h`.

3.47.2.22 `template<typename _Tpnu, typename _Tp> __gnu_cxx::__promote_2<_Tpnu, _Tp>::__type std::cyl_bessel_j (_Tpnu __nu, _Tp __x) [inline]`

Return the Bessel function $J_\nu(x)$ of real order ν and argument $x \geq 0$.

The cylindrical Bessel function is:

$$J_\nu(x) = \sum_{k=0}^{\infty} \frac{(-1)^k (x/2)^{\nu+2k}}{k! \Gamma(\nu + k + 1)}$$

Template Parameters

<code>_Tpnu</code>	The floating-point type of the order <code>__nu</code> .
<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .

Parameters

<code>__nu</code>	The order
<code>__x</code>	The argument, <code>__x >= 0</code>

Exceptions

<code>std::domain_error</code>	if <code>__x < 0</code> .
--------------------------------	------------------------------

Definition at line 581 of file `specfun.h`.

3.47.2.23 `float std::cyl_bessel_jf (float __nu, float __x) [inline]`

Return the Bessel function of the first kind $J_\nu(x)$ for `float` order ν and argument $x \geq 0$.

See also

`cyl_bessel_j` for details.

Definition at line 550 of file `specfun.h`.

3.47.2.24 `long double std::cyl_bessel_jl (long double __nu, long double __x) [inline]`

Return the Bessel function of the first kind $J_\nu(x)$ for `long double` order ν and argument $x \geq 0$.

See also

`cyl_bessel_j` for details.

Definition at line 560 of file `specfun.h`.

3.47.2.25 `template<typename _Tpnu, typename _Tp> __gnu_cxx::__promote_2<_Tpnu, _Tp>::__type std::cyl_bessel_k (_Tpnu __nu, _Tp __x) [inline]`

Return the irregular modified Bessel function $K_\nu(x)$ of real order ν and argument x .

The irregular modified Bessel function is defined by:

$$K_\nu(x) = \frac{\pi}{2} \frac{I_{-\nu}(x) - I_\nu(x)}{\sin \nu\pi}$$

where for integral $\nu = n$ a limit is taken: $\lim_{\nu \rightarrow n}$. For negative argument we have simply:

$$K_{-\nu}(x) = K_\nu(x)$$

Template Parameters

<code>_Tpnu</code>	The floating-point type of the order <code>__nu</code> .
<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .

Parameters

<code>__nu</code>	The order
<code>__x</code>	The argument, <code>__x >= 0</code>

Exceptions

<code>std::domain_error</code>	if <code>__x < 0</code> .
--------------------------------	------------------------------

Definition at line 633 of file `specfun.h`.

3.47.2.26 `float std::cyl_bessel_kf(float __nu, float __x) [inline]`

Return the irregular modified Bessel function $K_\nu(x)$ for `float` order ν and argument $x \geq 0$.

See also

`cyl_bessel_k` for setails.

Definition at line 596 of file `specfun.h`.

3.47.2.27 `long double std::cyl_bessel_kl(long double __nu, long double __x) [inline]`

Return the irregular modified Bessel function $K_\nu(x)$ for `long double` order ν and argument $x \geq 0$.

See also

`cyl_bessel_k` for setails.

Definition at line 606 of file `specfun.h`.

3.47.2.28 `template<typename _Tpnu, typename _Tp> __gnu_cxx::__promote_2<_Tpnu, _Tp>::__type std::cyl_neumann(_Tpnu __nu, _Tp __x) [inline]`

Return the Neumann function $N_\nu(x)$ of real order ν and argument $x \geq 0$.

The Neumann function is defined by:

$$N_\nu(x) = \frac{J_\nu(x) \cos \nu\pi - J_{-\nu}(x)}{\sin \nu\pi}$$

where $x \geq 0$ and for integral order $\nu = n$ a limit is taken: $\lim_{\nu \rightarrow n}$.

Template Parameters

<code>_Tpnu</code>	The floating-point type of the order <code>__nu</code> .
<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .

Parameters

<code>__nu</code>	The order
<code>__x</code>	The argument, <code>__x >= 0</code>

Exceptions

<code>std::domain_error</code>	if <code>__x < 0</code> .
--------------------------------	------------------------------

Definition at line 681 of file `specfun.h`.

3.47.2.29 `float std::cyl_neumannf (float __nu, float __x) [inline]`

Return the Neumann function $N_\nu(x)$ of `float` order ν and argument x .

See also

`cyl_neumann` for setails.

Definition at line 648 of file `specfun.h`.

3.47.2.30 `long double std::cyl_neumannl (long double __nu, long double __x) [inline]`

Return the Neumann function $N_\nu(x)$ of `long double` order ν and argument x .

See also

`cyl_neumann` for setails.

Definition at line 658 of file `specfun.h`.

3.47.2.31 `template<typename _Tp, typename _Tpp> __gnu_cxx::__promote_2<_Tp, _Tpp>::__type std::ellint_1 (_Tp __k, _Tpp __phi) [inline]`

Return the incomplete elliptic integral of the first kind $F(k, \phi)$ for `real` modulus k and angle ϕ .

The incomplete elliptic integral of the first kind is defined as

$$F(k, \phi) = \int_0^\phi \frac{d\theta}{\sqrt{1 - k^2 \sin^2 \theta}}$$

For $\phi = \pi/2$ this becomes the complete elliptic integral of the first kind, $K(k)$.

See also

`comp_ellint_1`.

Template Parameters

<code>_Tp</code>	The floating-point type of the modulus <code>__k</code> .
<code>_Tpp</code>	The floating-point type of the angle <code>__phi</code> .

Parameters

<code>__k</code>	The modulus, <code>abs (__k) <= 1</code>
<code>__phi</code>	The integral limit argument in radians

Exceptions

<code>std::domain_error</code>	if <code>abs (__k) > 1</code> .
--------------------------------	------------------------------------

Definition at line 729 of file `specfun.h`.

3.47.2.32 `float std::ellint_1f (float __k, float __phi) [inline]`

Return the incomplete elliptic integral of the first kind $E(k, \phi)$ for `float` modulus k and angle ϕ .

See also

`ellint_1` for details.

Definition at line 696 of file `specfun.h`.

3.47.2.33 `long double std::ellint_1l (long double __k, long double __phi) [inline]`

Return the incomplete elliptic integral of the first kind $E(k, \phi)$ for `long double` modulus k and angle ϕ .

See also

`ellint_1` for details.

Definition at line 706 of file `specfun.h`.

3.47.2.34 `template<typename _Tp, typename _Tpp> __gnu_cxx::__promote_2<_Tp, _Tpp>::__type std::ellint_2 (_Tp __k, _Tpp __phi) [inline]`

Return the incomplete elliptic integral of the second kind $E(k, \phi)$.

The incomplete elliptic integral of the second kind is defined as

$$E(k, \phi) = \int_0^\phi \sqrt{1 - k^2 \sin^2 \theta} d\theta$$

For $\phi = \pi/2$ this becomes the complete elliptic integral of the second kind, $E(k)$.

See also

`comp_ellint_2`.

Template Parameters

<code>__Tp</code>	The floating-point type of the modulus <code>__k</code> .
<code>__Tpp</code>	The floating-point type of the angle <code>__phi</code> .

Parameters

<code>__k</code>	The modulus, <code>abs (__k) <= 1</code>
<code>__phi</code>	The integral limit argument in radians

Returns

The elliptic function of the second kind.

Exceptions

<code>std::domain_error</code>	if <code>abs (__k) > 1</code> .
--------------------------------	------------------------------------

Definition at line 777 of file `specfun.h`.

3.47.2.35 `float std::ellint_2f (float __k, float __phi) [inline]`

Return the incomplete elliptic integral of the second kind $E(k, \phi)$ for `float` argument.

See also

`ellint_2` for details.

Definition at line 744 of file `specfun.h`.

3.47.2.36 `long double std::ellint_2l (long double __k, long double __phi) [inline]`

Return the incomplete elliptic integral of the second kind $E(k, \phi)$.

See also

`ellint_2` for details.

Definition at line 754 of file `specfun.h`.

3.47.2.37 `template<typename _Tp , typename _Tpn , typename _Tpp > __gnu_cxx::__promote_3<_Tp, _Tpn, _Tpp>::__type
std::ellint_3 (_Tp __k, _Tpn __nu, _Tpp __phi) [inline]`

Return the incomplete elliptic integral of the third kind $\Pi(k, \nu, \phi)$.

The incomplete elliptic integral of the third kind is defined by:

$$\Pi(k, \nu, \phi) = \int_0^\phi \frac{d\theta}{(1 - \nu \sin^2 \theta) \sqrt{1 - k^2 \sin^2 \theta}}$$

For $\phi = \pi/2$ this becomes the complete elliptic integral of the third kind, $\Pi(k, \nu)$.

See also

`comp_ellint_3`.

Template Parameters

<code>_Tp</code>	The floating-point type of the modulus <code>__k</code> .
<code>_Tpn</code>	The floating-point type of the argument <code>__nu</code> .
<code>_Tpp</code>	The floating-point type of the angle <code>__phi</code> .

Parameters

<code>__k</code>	The modulus, <code>abs (__k) <= 1</code>
<code>__nu</code>	The second argument
<code>__phi</code>	The integral limit argument in radians

Returns

The elliptic function of the third kind.

Exceptions

<code>std::domain_error</code>	if <code>abs (__k) > 1</code> .
--------------------------------	------------------------------------

Definition at line 830 of file `specfun.h`.

3.47.2.38 `float std::ellint_3f (float __k, float __nu, float __phi) [inline]`

Return the incomplete elliptic integral of the third kind $\Pi(k, \nu, \phi)$ for `float` argument.

See also

`ellint_3` for details.

Definition at line 792 of file `specfun.h`.

3.47.2.39 `long double std::ellint_3l (long double __k, long double __nu, long double __phi) [inline]`

Return the incomplete elliptic integral of the third kind $\Pi(k, \nu, \phi)$.

See also

`ellint_3` for details.

Definition at line 802 of file `specfun.h`.

3.47.2.40 `template<typename _Tp> __gnu_cxx::__promote<_Tp>::__type std::expint (_Tp __x) [inline]`

Return the exponential integral $Ei(x)$ for `real` argument `x`.

The exponential integral is given by

$$Ei(x) = - \int_{-x}^{\infty} \frac{e^t}{t} dt$$

Template Parameters

<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .
------------------	--

Parameters

<code>__x</code>	The argument of the exponential integral function.
------------------	--

Definition at line 870 of file `specfun.h`.

3.47.2.41 `float std::expintf (float __x) [inline]`

Return the exponential integral $Ei(x)$ for `float` argument `x`.

See also

`expint` for details.

Definition at line 844 of file `specfun.h`.

3.47.2.42 `long double std::expintl (long double __x) [inline]`

Return the exponential integral $Ei(x)$ for `long double` argument `x`.

See also

`expint` for details.

Definition at line 854 of file `specfun.h`.

3.47.2.43 `template<typename _Tp> __gnu_cxx::__promote<_Tp>::__type std::hermite (unsigned int __n, _Tp __x) [inline]`

Return the Hermite polynomial $H_n(x)$ of order `n` and `real` argument `x`.

The Hermite polynomial is defined by:

$$H_n(x) = (-1)^n e^{x^2} \frac{d^n}{dx^n} e^{-x^2}$$

The Hermite polynomial obeys a reflection formula:

$$H_n(-x) = (-1)^n H_n(x)$$

Template Parameters

<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .
------------------	--

Parameters

<code>__n</code>	The order
<code>__x</code>	The argument

Definition at line 918 of file `specfun.h`.

3.47.2.44 `float std::hermitef (unsigned int __n, float __x) [inline]`

Return the Hermite polynomial $H_n(x)$ of nonnegative order `n` and float argument `x`.

See also

`hermite` for details.

Definition at line 885 of file `specfun.h`.

3.47.2.45 `long double std::hermitel (unsigned int __n, long double __x) [inline]`

Return the Hermite polynomial $H_n(x)$ of nonnegative order `n` and `long double` argument `x`.

See also

`hermite` for details.

Definition at line 895 of file `specfun.h`.

3.47.2.46 `template<typename _Tp > __gnu_cxx::__promote<_Tp>::__type std::laguerre (unsigned int __n, _Tp __x) [inline]`

Returns the Laguerre polynomial $L_n(x)$ of nonnegative degree `n` and real argument $x \geq 0$.

The Laguerre polynomial is defined by:

$$L_n(x) = \frac{e^x}{n!} \frac{d^n}{dx^n} (x^n e^{-x})$$

Template Parameters

<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .
------------------	--

Parameters

<code>__n</code>	The nonnegative order
<code>__x</code>	The argument <code>__x >= 0</code>

Exceptions

<code>std::domain_error</code>	if <code>__x < 0</code> .
--------------------------------	------------------------------

Definition at line 962 of file `specfun.h`.

3.47.247 `float std::laguerref (unsigned int __n, float __x) [inline]`

Returns the Laguerre polynomial $L_n(x)$ of nonnegative degree `n` and `float` argument $x \geq 0$.

See also

`laguerre` for more details.

Definition at line 933 of file `specfun.h`.

3.47.248 `long double std::laguerrel (unsigned int __n, long double __x) [inline]`

Returns the Laguerre polynomial $L_n(x)$ of nonnegative degree `n` and `long double` argument $x \geq 0$.

See also

`laguerre` for more details.

Definition at line 943 of file `specfun.h`.

3.47.249 `template<typename _Tp> __gnu_cxx::__promote<_Tp>::__type std::legendre (unsigned int __l, _Tp __x) [inline]`

Return the Legendre polynomial $P_l(x)$ of nonnegative degree l and real argument $|x| \leq 0$.

The Legendre function of order l and argument x , $P_l(x)$, is defined by:

$$P_l(x) = \frac{1}{2^l l!} \frac{d^l}{dx^l} (x^2 - 1)^l$$

Template Parameters

<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .
------------------	--

Parameters

$_l$	The degree $l \geq 0$
$_x$	The argument $\text{abs}(_x) \leq 1$

Exceptions

<code>std::domain_error</code>	if $\text{abs}(_x) > 1$
--------------------------------	--------------------------

Definition at line 1007 of file specfun.h.

3.47.2.50 `float std::legendref (unsigned int __l, float __x)` `[inline]`

Return the Legendre polynomial $P_l(x)$ of nonnegative degree l and `float` argument $|x| \leq 0$.

See also

`legendre` for more details.

Definition at line 977 of file specfun.h.

3.47.2.51 `long double std::legendrel (unsigned int __l, long double __x)` `[inline]`

Return the Legendre polynomial $P_l(x)$ of nonnegative degree l and `long double` argument $|x| \leq 0$.

See also

`legendre` for more details.

Definition at line 987 of file specfun.h.

3.47.2.52 `template<typename _Tp> __gnu_cxx::__promote<_Tp>::__type std::riemann_zeta (_Tp __s)` `[inline]`

Return the Riemann zeta function $\zeta(s)$ for real argument s .

The Riemann zeta function is defined by:

$$\zeta(s) = \sum_{k=1}^{\infty} k^{-s} \text{ for } s > 1$$

and

$$\zeta(s) = \frac{1}{1 - 2^{1-s}} \sum_{k=1}^{\infty} (-1)^{k-1} k^{-s} \text{ for } 0 \leq s \leq 1$$

For $s < 1$ use the reflection formula:

$$\zeta(s) = 2^s \pi^{s-1} \sin\left(\frac{\pi s}{2}\right) \Gamma(1-s) \zeta(1-s)$$

Template Parameters

<code>_Tp</code>	The floating-point type of the argument <code>__s</code> .
------------------	--

Parameters

<code>__s</code>	The argument $s \neq 1$
------------------	-------------------------

Definition at line 1058 of file `specfun.h`.

3.47.2.53 `float std::riemann_zetaf(float __s) [inline]`

Return the Riemann zeta function $\zeta(s)$ for `float` argument s .

See also

`riemann_zeta` for more details.

Definition at line 1022 of file `specfun.h`.

3.47.2.54 `long double std::riemann_zetal(long double __s) [inline]`

Return the Riemann zeta function $\zeta(s)$ for `long double` argument s .

See also

`riemann_zeta` for more details.

Definition at line 1032 of file `specfun.h`.

3.47.2.55 `template<typename _Tp> __gnu_cxx::__promote<_Tp>::__type std::sph_bessel(unsigned int __n, _Tp __x) [inline]`

Return the spherical Bessel function $j_n(x)$ of nonnegative order n and real argument $x \geq 0$.

The spherical Bessel function is defined by:

$$j_n(x) = \left(\frac{\pi}{2x}\right)^{1/2} J_{n+1/2}(x)$$

Template Parameters

<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .
------------------	--

Parameters

\leftrightarrow _n	The integral order $n \geq 0$
\leftrightarrow _x	The real argument $x \geq 0$

Exceptions

<code>std::domain_error</code>	if <code>__x < 0</code> .
--------------------------------	------------------------------

Definition at line 1102 of file `specfun.h`.

3.47.2.56 `float std::sph_besself (unsigned int __n, float __x) [inline]`

Return the spherical Bessel function $j_n(x)$ of nonnegative order n and `float` argument $x \geq 0$.

See also

`sph_bessel` for more details.

Definition at line 1073 of file `specfun.h`.

3.47.2.57 `long double std::sph_bessell (unsigned int __n, long double __x) [inline]`

Return the spherical Bessel function $j_n(x)$ of nonnegative order n and `long double` argument $x \geq 0$.

See also

`sph_bessel` for more details.

Definition at line 1083 of file `specfun.h`.

3.47.2.58 `template<typename _Tp> __gnu_cxx::__promote<_Tp>::__type std::sph_legendre (unsigned int __l, unsigned int __m, _Tp __theta) [inline]`

Return the spherical Legendre function of nonnegative integral degree l and order m and real angle θ in radians.

The spherical Legendre function is defined by

$$Y_l^m(\theta, \phi) = (-1)^m \left[\frac{(2l+1)(l-m)!}{4\pi(l+m)!} \right] P_l^m(\cos \theta) \exp^{im\phi}$$

Template Parameters

<code>_Tp</code>	The floating-point type of the angle <code>__theta</code> .
------------------	---

Parameters

<code>__l</code>	The order <code>__l >= 0</code>
<code>__m</code>	The degree <code>__m >= 0</code> and <code>__m <= __l</code>
<code>__theta</code>	The radian polar angle argument

Definition at line 1149 of file `specfun.h`.

3.47.2.59 `float std::sph_legendref (unsigned int __l, unsigned int __m, float __theta) [inline]`

Return the spherical Legendre function of nonnegative integral degree `l` and order `m` and float angle θ in radians.

See also

`sph_legendre` for details.

Definition at line 1117 of file `specfun.h`.

3.47.2.60 `long double std::sph_legendrel (unsigned int __l, unsigned int __m, long double __theta) [inline]`

Return the spherical Legendre function of nonnegative integral degree `l` and order `m` and long double angle θ in radians.

See also

`sph_legendre` for details.

Definition at line 1128 of file `specfun.h`.

3.47.2.61 `template<typename _Tp> __gnu_cxx::__promote<_Tp>::__type std::sph_neumann (unsigned int __n, _Tp __x) [inline]`

Return the spherical Neumann function of integral order $n \geq 0$ and real argument $x \geq 0$.

The spherical Neumann function is defined by

$$n_n(x) = \left(\frac{\pi}{2x}\right)^{1/2} N_{n+1/2}(x)$$

Template Parameters

<code>_Tp</code>	The floating-point type of the argument <code>__x</code> .
------------------	--

Parameters

\leftrightarrow _n	The integral order $n \geq 0$
\leftrightarrow _x	The real argument $__x \geq 0$

Exceptions

<code>std::domain_error</code>	if $__x < 0$.
--------------------------------	----------------

Definition at line 1193 of file specfun.h.

3.47.2.62 `float std::sph_neumannf (unsigned int __n, float __x) [inline]`

Return the spherical Neumann function of integral order $n \geq 0$ and `float` argument $x \geq 0$.

See also

`sph_neumann` for details.

Definition at line 1164 of file specfun.h.

3.47.2.63 `long double std::sph_neumannl (unsigned int __n, long double __x) [inline]`

Return the spherical Neumann function of integral order $n \geq 0$ and `long double` $x \geq 0$.

See also

`sph_neumann` for details.

Definition at line 1174 of file specfun.h.

3.48 Mathematical Special Functions

Collaboration diagram for Mathematical Special Functions:



Functions

- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::tr1::assoc_laguerre` (unsigned int __n, unsigned int __m, _Tp __x)
- `float std::tr1::assoc_laguerref` (unsigned int __n, unsigned int __m, float __x)
- `long double std::tr1::assoc_laguerrel` (unsigned int __n, unsigned int __m, long double __x)
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::tr1::assoc_legendre` (unsigned int __l, unsigned int __m, _Tp __x)
- `float std::tr1::assoc_legendref` (unsigned int __l, unsigned int __m, float __x)
- `long double std::tr1::assoc_legendrel` (unsigned int __l, unsigned int __m, long double __x)
- `template<typename _Tpx, typename _Tpy >`
`__gnu_cxx::__promote_2< _Tpx, _Tpy >::__type std::tr1::beta` (_Tpx __x, _Tpy __y)
- `float std::tr1::betaf` (float __x, float __y)
- `long double std::tr1::betal` (long double __x, long double __y)
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::tr1::comp_ellint_1` (_Tp __k)
- `float std::tr1::comp_ellint_1f` (float __k)
- `long double std::tr1::comp_ellint_1l` (long double __k)
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::tr1::comp_ellint_2` (_Tp __k)
- `float std::tr1::comp_ellint_2f` (float __k)
- `long double std::tr1::comp_ellint_2l` (long double __k)
- `template<typename _Tp, typename _Tpn >`
`__gnu_cxx::__promote_2< _Tp, _Tpn >::__type std::tr1::comp_ellint_3` (_Tp __k, _Tpn __nu)
- `float std::tr1::comp_ellint_3f` (float __k, float __nu)
- `long double std::tr1::comp_ellint_3l` (long double __k, long double __nu)
- `template<typename _Tpa, typename _Tpc, typename _Tp >`
`__gnu_cxx::__promote_3< _Tpa, _Tpc, _Tp >::__type std::tr1::conf_hyperg` (_Tpa __a, _Tpc __c, _Tp __x)
- `float std::tr1::conf_hypergf` (float __a, float __c, float __x)
- `long double std::tr1::conf_hypergl` (long double __a, long double __c, long double __x)
- `template<typename _Tpnu, typename _Tp >`
`__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::tr1::cyl_bessel_i` (_Tpnu __nu, _Tp __x)
- `float std::tr1::cyl_bessel_if` (float __nu, float __x)
- `long double std::tr1::cyl_bessel_il` (long double __nu, long double __x)
- `template<typename _Tpnu, typename _Tp >`
`__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::tr1::cyl_bessel_j` (_Tpnu __nu, _Tp __x)

- float **std::tr1::cyl_bessel_jf** (float __nu, float __x)
- long double **std::tr1::cyl_bessel_jl** (long double __nu, long double __x)
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type **std::tr1::cyl_bessel_k** (_Tpnu __nu, _Tp __x)
- float **std::tr1::cyl_bessel_kf** (float __nu, float __x)
- long double **std::tr1::cyl_bessel_kl** (long double __nu, long double __x)
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type **std::tr1::cyl_neumann** (_Tpnu __nu, _Tp __x)
- float **std::tr1::cyl_neumannf** (float __nu, float __x)
- long double **std::tr1::cyl_neumannl** (long double __nu, long double __x)
- template<typename _Tp, typename _Tpp >
__gnu_cxx::__promote_2< _Tp, _Tpp >::__type **std::tr1::ellint_1** (_Tp __k, _Tpp __phi)
- float **std::tr1::ellint_1f** (float __k, float __phi)
- long double **std::tr1::ellint_1l** (long double __k, long double __phi)
- template<typename _Tp, typename _Tpp >
__gnu_cxx::__promote_2< _Tp, _Tpp >::__type **std::tr1::ellint_2** (_Tp __k, _Tpp __phi)
- float **std::tr1::ellint_2f** (float __k, float __phi)
- long double **std::tr1::ellint_2l** (long double __k, long double __phi)
- template<typename _Tp, typename _Tpn, typename _Tpp >
__gnu_cxx::__promote_3< _Tp, _Tpn, _Tpp >::__type **std::tr1::ellint_3** (_Tp __k, _Tpn __nu, _Tpp __phi)
- float **std::tr1::ellint_3f** (float __k, float __nu, float __phi)
- long double **std::tr1::ellint_3l** (long double __k, long double __nu, long double __phi)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::expint** (_Tp __x)
- float **std::tr1::expintf** (float __x)
- long double **std::tr1::expintl** (long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::hermite** (unsigned int __n, _Tp __x)
- float **std::tr1::hermitef** (unsigned int __n, float __x)
- long double **std::tr1::hermitel** (unsigned int __n, long double __x)
- template<typename _Tpa, typename _Tpb, typename _Tpc, typename _Tp >
__gnu_cxx::__promote_4< _Tpa, _Tpb, _Tpc, _Tp >::__type **std::tr1::hyperg** (_Tpa __a, _Tpb __b, _Tpc __c, _Tp __x)
- float **std::tr1::hypergf** (float __a, float __b, float __c, float __x)
- long double **std::tr1::hypergl** (long double __a, long double __b, long double __c, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::laguerre** (unsigned int __n, _Tp __x)
- float **std::tr1::laguerref** (unsigned int __n, float __x)
- long double **std::tr1::laguerrel** (unsigned int __n, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::legendre** (unsigned int __n, _Tp __x)
- float **std::tr1::legendref** (unsigned int __n, float __x)
- long double **std::tr1::legendrel** (unsigned int __n, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::riemann_zeta** (_Tp __x)
- float **std::tr1::riemann_zetaf** (float __x)
- long double **std::tr1::riemann_zetal** (long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::sph_bessel** (unsigned int __n, _Tp __x)
- float **std::tr1::sph_besself** (unsigned int __n, float __x)
- long double **std::tr1::sph_bessell** (unsigned int __n, long double __x)

- `template<typename _Tp > __gnu_cxx::__promote< _Tp >::__type std::tr1::sph_legendre (unsigned int __l, unsigned int __m, _Tp __theta)`
- `float std::tr1::sph_legendref (unsigned int __l, unsigned int __m, float __theta)`
- `long double std::tr1::sph_legendrel (unsigned int __l, unsigned int __m, long double __theta)`
- `template<typename _Tp > __gnu_cxx::__promote< _Tp >::__type std::tr1::sph_neumann (unsigned int __n, _Tp __x)`
- `float std::tr1::sph_neumannf (unsigned int __n, float __x)`
- `long double std::tr1::sph_neumannl (unsigned int __n, long double __x)`

3.48.1 Detailed Description

A collection of advanced mathematical special functions.

3.48.2 Function Documentation

3.48.2.1 `template<typename _Tp > __gnu_cxx::__promote<_Tp>::__type std::tr1::assoc_laguerre (unsigned int __n, unsigned int __m, _Tp __x) [inline]`

5.2.1.1 Associated Laguerre polynomials.

Definition at line 1302 of file tr1/cmath.

3.48.2.2 `template<typename _Tp > __gnu_cxx::__promote<_Tp>::__type std::tr1::assoc_legendre (unsigned int __l, unsigned int __m, _Tp __x) [inline]`

5.2.1.2 Associated Legendre functions.

Definition at line 1319 of file tr1/cmath.

3.48.2.3 `template<typename _Tpx, typename _Tpy > __gnu_cxx::__promote_2<_Tpx, _Tpy>::__type std::tr1::beta (_Tpx __x, _Tpy __y) [inline]`

5.2.1.3 Beta functions.

Definition at line 1336 of file tr1/cmath.

3.48.2.4 `template<typename _Tp > __gnu_cxx::__promote<_Tp>::__type std::tr1::comp_ellint_1 (_Tp __k) [inline]`

5.2.1.4 Complete elliptic integrals of the first kind.

Definition at line 1353 of file tr1/cmath.

3.48.2.5 `template<typename _Tp > __gnu_cxx::__promote<_Tp>::__type std::tr1::comp_ellint_2 (_Tp __k) [inline]`

5.2.1.5 Complete elliptic integrals of the second kind.

Definition at line 1370 of file tr1/cmath.

3.48.2.6 `template<typename _Tp, typename _Tpn > __gnu_cxx::__promote_2<_Tp, _Tpn>::__type std::tr1::comp_ellint_3 (_Tp __k, _Tpn __nu) [inline]`

5.2.1.6 Complete elliptic integrals of the third kind.

Definition at line 1387 of file tr1/cmath.

3.48.2.7 `template<typename _Tpa, typename _Tpc, typename _Tp > __gnu_cxx::__promote_3<_Tpa, _Tpc, _Tp>::__type std::tr1::conf_hyperg (_Tpa __a, _Tpc __c, _Tp __x) [inline]`

5.2.1.7 Confluent hypergeometric functions.

Definition at line 1404 of file tr1/cmath.

3.48.2.8 `template<typename _Tpnu, typename _Tp > __gnu_cxx::__promote_2<_Tpnu, _Tp>::__type std::tr1::cyl_bessel_i (_Tpnu __nu, _Tp __x) [inline]`

5.2.1.8 Regular modified cylindrical Bessel functions.

Definition at line 1421 of file tr1/cmath.

3.48.2.9 `template<typename _Tpnu, typename _Tp > __gnu_cxx::__promote_2<_Tpnu, _Tp>::__type std::tr1::cyl_bessel_j (_Tpnu __nu, _Tp __x) [inline]`

5.2.1.9 Cylindrical Bessel functions (of the first kind).

Definition at line 1438 of file tr1/cmath.

3.48.2.10 `template<typename _Tpnu, typename _Tp > __gnu_cxx::__promote_2<_Tpnu, _Tp>::__type std::tr1::cyl_bessel_k (_Tpnu __nu, _Tp __x) [inline]`

5.2.1.10 Irregular modified cylindrical Bessel functions.

Definition at line 1455 of file tr1/cmath.

3.48.2.11 `template<typename _Tpnu, typename _Tp > __gnu_cxx::__promote_2<_Tpnu, _Tp>::__type std::tr1::cyl_neumann (_Tpnu __nu, _Tp __x) [inline]`

5.2.1.11 Cylindrical Neumann functions.

Definition at line 1472 of file tr1/cmath.

3.48.2.12 `template<typename _Tp, typename _Tpp > __gnu_cxx::__promote_2<_Tp, _Tpp>::__type std::tr1::ellint_1 (_Tp __k, _Tpp __phi) [inline]`

5.2.1.12 Incomplete elliptic integrals of the first kind.

Definition at line 1489 of file tr1/cmath.

3.48.2.13 `template<typename _Tp, typename _Tpp> __gnu_cxx::__promote_2<_Tp, _Tpp>::__type std::tr1::ellint_2 (_Tp __k, _Tpp __phi) [inline]`

5.2.1.13 Incomplete elliptic integrals of the second kind.

Definition at line 1506 of file tr1/cmath.

3.48.2.14 `template<typename _Tp, typename _Tpn, typename _Tpp> __gnu_cxx::__promote_3<_Tp, _Tpn, _Tpp>::__type std::tr1::ellint_3 (_Tp __k, _Tpn __nu, _Tpp __phi) [inline]`

5.2.1.14 Incomplete elliptic integrals of the third kind.

Definition at line 1523 of file tr1/cmath.

3.48.2.15 `template<typename _Tp> __gnu_cxx::__promote<_Tp>::__type std::tr1::expint (_Tp __x) [inline]`

5.2.1.15 Exponential integrals.

Definition at line 1540 of file tr1/cmath.

3.48.2.16 `template<typename _Tp> __gnu_cxx::__promote<_Tp>::__type std::tr1::hermite (unsigned int __n, _Tp __x) [inline]`

5.2.1.16 Hermite polynomials.

Definition at line 1557 of file tr1/cmath.

3.48.2.17 `template<typename _Tpa, typename _Tpb, typename _Tpc, typename _Tp> __gnu_cxx::__promote_4<_Tpa, _Tpb, _Tpc, _Tp>::__type std::tr1::hyperg (_Tpa __a, _Tpb __b, _Tpc __c, _Tp __x) [inline]`

5.2.1.17 Hypergeometric functions.

Definition at line 1574 of file tr1/cmath.

3.48.2.18 `template<typename _Tp> __gnu_cxx::__promote<_Tp>::__type std::tr1::laguerre (unsigned int __n, _Tp __x) [inline]`

5.2.1.18 Laguerre polynomials.

Definition at line 1591 of file tr1/cmath.

3.48.2.19 `template<typename _Tp> __gnu_cxx::__promote<_Tp>::__type std::tr1::legendre (unsigned int __n, _Tp __x) [inline]`

5.2.1.19 Legendre polynomials.

Definition at line 1608 of file tr1/cmath.

3.48.2.20 `template<typename _Tp> __gnu_cxx::__promote<_Tp>::__type std::tr1::riemann_zeta (_Tp __x) [inline]`

5.2.1.20 Riemann zeta function.

Definition at line 1625 of file tr1/cmath.

3.48.2.21 `template<typename _Tp> __gnu_cxx::__promote<_Tp>::__type std::tr1::sph_bessel (unsigned int __n, _Tp __x) [inline]`

5.2.1.21 Spherical Bessel functions.

Definition at line 1642 of file tr1/cmath.

3.48.2.22 `template<typename _Tp> __gnu_cxx::__promote<_Tp>::__type std::tr1::sph_legendre (unsigned int __l, unsigned int __m, _Tp __theta) [inline]`

5.2.1.22 Spherical associated Legendre functions.

Definition at line 1659 of file tr1/cmath.

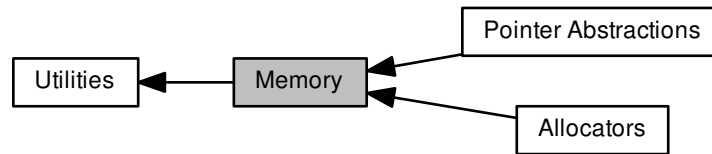
3.48.2.23 `template<typename _Tp> __gnu_cxx::__promote<_Tp>::__type std::tr1::sph_neumann (unsigned int __n, _Tp __x) [inline]`

5.2.1.23 Spherical Neumann functions.

Definition at line 1676 of file tr1/cmath.

3.49 Memory

Collaboration diagram for Memory:



Modules

- [Allocators](#)
- [Pointer Abstractions](#)

3.49.1 Detailed Description

Components for memory allocation, deallocation, and management.

3.50 Metaprogramming

Collaboration diagram for Metaprogramming:



Classes

- struct `std::__is_nullptr_t< _Tp >`
- struct `std::integral_constant< _Tp, __v >`
- struct `std::is_abstract< _Tp >`
- struct `std::is_arithmetic< _Tp >`
- struct `std::is_array< typename >`
- struct `std::is_class< _Tp >`
- struct `std::is_compound< _Tp >`
- struct `std::is_const< typename >`
- struct `std::is_empty< _Tp >`
- struct `std::is_enum< _Tp >`
- struct `std::is_final< _Tp >`
- struct `std::is_floating_point< _Tp >`
- struct `std::is_function< typename >`
- struct `std::is_fundamental< _Tp >`
- struct `std::is_integral< _Tp >`
- struct `std::is_literal_type< _Tp >`
- struct `std::is_lvalue_reference< typename >`
- struct `std::is_member_function_pointer< _Tp >`
- struct `std::is_member_object_pointer< _Tp >`
- struct `std::is_member_pointer< _Tp >`
- struct `std::is_null_pointer< _Tp >`
- struct `std::is_object< _Tp >`
- struct `std::is_pod< _Tp >`
- struct `std::is_pointer< _Tp >`
- struct `std::is_polymorphic< _Tp >`
- struct `std::is_reference< _Tp >`
- struct `std::is_rvalue_reference< typename >`
- struct `std::is_scalar< _Tp >`
- struct `std::is_standard_layout< _Tp >`
- struct `std::is_trivial< _Tp >`
- struct `std::is_union< _Tp >`
- struct `std::is_void< _Tp >`
- struct `std::is_volatile< typename >`
- struct `std::tr2::__reflection_typelist< _Elements >`
- struct `std::tr2::__reflection_typelist< _First, _Rest... >`
- struct `std::tr2::__reflection_typelist<>`
- struct `std::tr2::bases< _Tp >`
- struct `std::tr2::direct_bases< _Tp >`

Macros

- `#define __cpp_lib_integral_constant_callable`
- `#define __cpp_lib_is_final`
- `#define __cpp_lib_is_null_pointer`

Typedefs

- `template<bool __v>`
`using std::__bool_constant = integral_constant< bool, __v >`
- `typedef integral_constant< bool, false > std::false_type`
- `typedef integral_constant< bool, true > std::true_type`
- `typedef integral_constant< _Tp, __v > std::integral_constant< _Tp, __v >::type`
- `typedef _Tp std::__success_type< _Tp >::type`
- `typedef _Tp std::integral_constant< _Tp, __v >::value_type`

Functions

- `std::__nonesuch::__nonesuch (__nonesuch const &)=delete`
- `constexpr std::integral_constant< _Tp, __v >::operator value_type () const`
- `constexpr value_type std::integral_constant< _Tp, __v >::operator() () const`
- `void std::__nonesuch::operator= (__nonesuch const &)=delete`

Variables

- `static constexpr _Tp std::integral_constant< _Tp, __v >::value`

3.50.1 Detailed Description

Template utilities for compile-time introspection and modification, including type classification traits, type property inspection traits and type transformation traits.

3.50.2 Typedef Documentation

3.50.2.1 `typedef integral_constant<bool, false> std::false_type`

The type used as a compile-time boolean with false value.

Definition at line 90 of file `type_traits`.

3.50.2.2 `typedef integral_constant<bool, true> std::true_type`

The type used as a compile-time boolean with true value.

Definition at line 87 of file `type_traits`.

3.51 Mutating

Collaboration diagram for Mutating:



Functions

- `template<typename _II, typename _OI >`
`_OI std::copy (_II __first, _II __last, _OI __result)`
- `template<typename _BI1, typename _BI2 >`
`_BI2 std::copy_backward (_BI1 __first, _BI1 __last, _BI2 __result)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Predicate __pred)`
- `template<typename _InputIterator, typename _Size, typename _OutputIterator >`
`_OutputIterator std::copy_n (_InputIterator __first, _Size __n, _OutputIterator __result)`
- `template<typename _ForwardIterator, typename _Tp >`
`void std::fill (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value)`
- `template<typename _OI, typename _Size, typename _Tp >`
`_OI std::fill_n (_OI __first, _Size __n, const _Tp &__value)`
- `template<typename _ForwardIterator, typename _Generator >`
`void std::generate (_ForwardIterator __first, _ForwardIterator __last, _Generator __gen)`
- `template<typename _OutputIterator, typename _Size, typename _Generator >`
`_OutputIterator std::generate_n (_OutputIterator __first, _Size __n, _Generator __gen)`
- `template<typename _InputIterator, typename _Predicate >`
`bool std::is_partitioned (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`void std::iter_swap (_ForwardIterator1 __a, _ForwardIterator2 __b)`
- `template<typename _II, typename _OI >`
`_OI std::move (_II __first, _II __last, _OI __result)`
- `template<typename _BI1, typename _BI2 >`
`_BI2 std::move_backward (_BI1 __first, _BI1 __last, _BI2 __result)`
- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator std::partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _OutputIterator1, typename _OutputIterator2, typename _Predicate >`
`pair< _OutputIterator1, _OutputIterator2 > std::partition_copy (_InputIterator __first, _InputIterator __last, __<_OutputIterator1 __out_true, _OutputIterator2 __out_false, _Predicate __pred)`
- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator std::partition_point (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _RandomAccessIterator, typename _RandomNumberGenerator >`
`void std::random_shuffle (_RandomAccessIterator __first, _RandomAccessIterator __last, _RandomNumberGenerator &&__rand)`

- `template<typename _ForwardIterator, typename _Tp >`
`_ForwardIterator std::remove (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Tp >`
`_OutputIterator std::remove_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result, const _Tp &__value)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::remove_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Pred Predicate __pred)`
- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator std::remove_if (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _ForwardIterator, typename _Tp >`
`void std::replace (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__old_value, const _Tp &__new_value)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate, typename _Tp >`
`_OutputIterator std::replace_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Pred Predicate __pred, const _Tp &__new_value)`
- `template<typename _ForwardIterator, typename _Predicate, typename _Tp >`
`void std::replace_if (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred, const _Tp &__new_value)`
- `template<typename _BidirectionalIterator >`
`void std::reverse (_BidirectionalIterator __first, _BidirectionalIterator __last)`
- `template<typename _BidirectionalIterator, typename _OutputIterator >`
`_OutputIterator std::reverse_copy (_BidirectionalIterator __first, _BidirectionalIterator __last, _OutputIterator __result)`
- `template<typename _ForwardIterator >`
`_ForwardIterator std::V2::rotate (_ForwardIterator __first, _ForwardIterator __middle, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _OutputIterator >`
`_OutputIterator std::rotate_copy (_ForwardIterator __first, _ForwardIterator __middle, _ForwardIterator __last, _OutputIterator __result)`
- `template<typename _RandomAccessIterator, typename _UniformRandomNumberGenerator >`
`void std::shuffle (_RandomAccessIterator __first, _RandomAccessIterator __last, _UniformRandomNumberGenerator &&__g)`
- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator std::stable_partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`_ForwardIterator2 std::swap_ranges (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2)`
- `template<typename _InputIterator, typename _OutputIterator, typename _UnaryOperation >`
`_OutputIterator std::transform (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _UnaryOperation __unary_op)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::transform (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _OutputIterator __result, _BinaryOperation __binary_op)`
- `template<typename _ForwardIterator >`
`_ForwardIterator std::unique (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _BinaryPredicate >`
`_ForwardIterator std::unique (_ForwardIterator __first, _ForwardIterator __last, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator, typename _OutputIterator >`
`_OutputIterator std::unique_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryPredicate >`
`_OutputIterator std::unique_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _BinaryPredicate __binary_pred)`

3.51.1 Detailed Description

3.51.2 Function Documentation

3.51.2.1 `template<typename _II, typename _OI> _OI std::copy (_II __first, _II __last, _OI __result) [inline]`

Copies the range [first,last) into result.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.

Returns

`result + (first - last)`

This inline function will boil down to a call to `memmove` whenever possible. Failing that, if random access iterators are passed, then the loop count will be known (and therefore a candidate for compiler optimizations such as unrolling). Result may not be contained within [first,last); the `copy_backward` function should be used instead.

Note that the end of the output range is permitted to be contained within [first,last).

Definition at line 446 of file `stl_algobase.h`.

3.51.2.2 `template<typename _BI1, typename _BI2> _BI2 std::copy_backward (_BI1 __first, _BI1 __last, _BI2 __result) [inline]`

Copies the range [first,last) into result.

Parameters

<code>__first</code>	A bidirectional iterator.
<code>__last</code>	A bidirectional iterator.
<code>__result</code>	A bidirectional iterator.

Returns

`result - (first - last)`

The function has the same effect as `copy`, but starts at the end of the range and works its way to the start, returning the start of the result. This inline function will boil down to a call to `memmove` whenever possible. Failing that, if random access iterators are passed, then the loop count will be known (and therefore a candidate for compiler optimizations such as unrolling).

Result may not be in the range (first,last]. Use `copy` instead. Note that the start of the output range may overlap [first,last).

Definition at line 622 of file `stl_algobase.h`.

3.51.2.3 `template<typename _InputIterator, typename _OutputIterator, typename _Predicate > _OutputIterator std::copy_if (`
`_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Predicate __pred)`

Copy the elements of a sequence for which a predicate is true.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.
<code>__pred</code>	A predicate.

Returns

An iterator designating the end of the resulting sequence.

Copies each element in the range `[__first,__last)` for which `__pred` returns true to the range beginning at `__result`.

`copy_if()` is stable, so the relative order of elements that are copied is unchanged.

Definition at line 734 of file `stl_algo.h`.

3.51.2.4 `template<typename _InputIterator, typename _Size, typename _OutputIterator > _OutputIterator std::copy_n (`
`_InputIterator __first, _Size __n, _OutputIterator __result) [inline]`

Copies the range `[first,first+n)` into `[result,result+n)`.

Parameters

<code>__first</code>	An input iterator.
<code>__n</code>	The number of elements to copy.
<code>__result</code>	An output iterator.

Returns

`result+n`.

This inline function will boil down to a call to `memmove` whenever possible. Failing that, if random access iterators are passed, then the loop count will be known (and therefore a candidate for compiler optimizations such as unrolling).

Definition at line 796 of file `stl_algo.h`.

References `std::__iterator_category()`.

3.51.2.5 `template<typename _ForwardIterator, typename _Tp > void std::fill (_ForwardIterator __first, _ForwardIterator __last,`
`const _Tp & __value) [inline]`

Fills the range `[first,last)` with copies of `value`.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__value</code>	A reference-to-const of arbitrary type.

Returns

Nothing.

This function fills a range with copies of the same value. For char types filling contiguous areas of memory, this becomes an inline call to `memset` or `wmemset`.

Definition at line 724 of file `stl_algobase.h`.

```
3.51.2.6 template<typename _OI, typename _Size, typename _Tp> _OI std::fill_n ( _OI __first, _Size __n, const _Tp & __value )
[inline]
```

Fills the range `[first,first+n)` with copies of value.

Parameters

<code>__first</code>	An output iterator.
<code>__n</code>	The count of copies to perform.
<code>__value</code>	A reference-to-const of arbitrary type.

Returns

The iterator at `first+n`.

This function fills a range with copies of the same value. For char types filling contiguous areas of memory, this becomes an inline call to `memset` or `@ wmemset`.

`_GLIBCXX_RESOLVE_LIB_DEFECTS` DR 865. More algorithms that throw away information

Definition at line 784 of file `stl_algobase.h`.

References `std::advance()`, `std::distance()`, `std::equal()`, and `std::min()`.

Referenced by `std::vector<_Tp, _Alloc>::operator=()`, `std::uninitialized_fill()`, and `std::uninitialized_fill_n()`.

```
3.51.2.7 template<typename _ForwardIterator, typename _Generator> void std::generate ( _ForwardIterator __first,
_FowardIterator __last, _Generator __gen )
```

Assign the result of a function object to each value in a sequence.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__gen</code>	A function object taking no arguments and returning <code>std::iterator_traits<_ForwardIterator>::value_type</code>

Returns

`generate()` returns no value.

Performs the assignment `*i = __gen()` for each `i` in the range `[__first, __last)`.

Definition at line 4299 of file `stl_algo.h`.

3.51.2.8 `template<typename _OutputIterator, typename _Size, typename _Generator> _OutputIterator std::generate_n (`
`_OutputIterator __first, _Size __n, _Generator __gen)`

Assign the result of a function object to each value in a sequence.

Parameters

<code>__first</code>	A forward iterator.
<code>__n</code>	The length of the sequence.
<code>__gen</code>	A function object taking no arguments and returning <code>std::iterator_traits<_ForwardIterator>::value_type</code>

Returns

The end of the sequence, `__first+__n`

Performs the assignment `*i = __gen()` for each `i` in the range `[__first, __first+__n)`.

`_GLIBCXX_RESOLVE_LIB_DEFECTS` DR 865. More algorithms that throw away information

Definition at line 4330 of file `stl_algo.h`.

3.51.2.9 `template<typename _InputIterator, typename _Predicate> bool std::is_partitioned (_InputIterator __first, _InputIterator`
`__last, _Predicate __pred) [inline]`

Checks whether the sequence is partitioned.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__pred</code>	A predicate.

Returns

True if the range `[__first,__last)` is partitioned by `__pred`, i.e. if all elements that satisfy `__pred` appear before those that do not.

Definition at line 582 of file `stl_algo.h`.

References `std::find_if_not()`, and `std::none_of()`.

3.51.2.10 `template<typename _ForwardIterator1, typename _ForwardIterator2> void std::iter_swap (_ForwardIterator1 __a, _ForwardIterator2 __b) [inline]`

Swaps the contents of two iterators.

Parameters

<code>↔ __a</code>	An iterator.
<code>↔ __b</code>	Another iterator.

Returns

Nothing.

This function swaps the values pointed to by two iterators, not the iterators themselves.

Definition at line 120 of file `stl_algobase.h`.

Referenced by `std::__introsort_loop()`, `std::__merge_without_buffer()`, `std::__move_median_to_first()`, `std::__partition()`, `std::__reverse()`, `std::__V2::__rotate()`, `std::__unguarded_partition()`, `std::includes()`, `std::next_permutation()`, `std::__random_shuffle()`, `std::shuffle()`, `std::swap_ranges()`, and `std::unique_copy()`.

3.51.2.11 `template<typename _II, typename _OI> _OI std::move (_II __first, _II __last, _OI __result) [inline]`

Moves the range `[first,last)` into `result`.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.

Returns

`result + (first - last)`

This inline function will boil down to a call to `memmove` whenever possible. Failing that, if random access iterators

are passed, then the loop count will be known (and therefore a candidate for compiler optimizations such as unrolling). Result may not be contained within `[first,last)`; the `move_backward` function should be used instead.

Note that the end of the output range is permitted to be contained within `[first,last)`.

Definition at line 479 of file `stl_algobase.h`.

3.51.2.12 `template<typename _BI1 , typename _BI2 > _BI2 std::move_backward (_BI1 __first, _BI1 __last, _BI2 __result)`
`[inline]`

Moves the range `[first,last)` into `result`.

Parameters

<code>__first</code>	A bidirectional iterator.
<code>__last</code>	A bidirectional iterator.
<code>__result</code>	A bidirectional iterator.

Returns

`result - (first - last)`

The function has the same effect as `move`, but starts at the end of the range and works its way to the start, returning the start of the result. This inline function will boil down to a call to `memmove` whenever possible. Failing that, if random access iterators are passed, then the loop count will be known (and therefore a candidate for compiler optimizations such as unrolling).

Result may not be in the range `(first,last]`. Use `move` instead. Note that the start of the output range may overlap `[first,last)`.

Definition at line 658 of file `stl_algobase.h`.

3.51.2.13 `template<typename _ForwardIterator , typename _Predicate > _ForwardIterator std::partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)` `[inline]`

Move elements for which a predicate is true to the beginning of a sequence.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__pred</code>	A predicate functor.

Returns

An iterator `middle` such that `__pred(i)` is true for each iterator `i` in the range `[__first,middle)` and false for each `i` in the range `[middle,__last)`.

`__pred` must not modify its operand. `partition()` does not preserve the relative ordering of elements in each group, use `stable_partition()` if this is needed.

Definition at line 4514 of file `stl_algo.h`.

References `std::__iterator_category()`, and `std::__partition()`.

```
3.51.2.14 template<typename _InputIterator, typename _OutputIterator1, typename _OutputIterator2, typename _Predicate
> pair<_OutputIterator1, _OutputIterator2> std::partition_copy ( _InputIterator __first, _InputIterator __last,
    _OutputIterator1 __out_true, _OutputIterator2 __out_false, _Predicate __pred )
```

Copy the elements of a sequence to separate output sequences depending on the truth value of a predicate.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__out_true</code>	An output iterator.
<code>__out_false</code>	An output iterator.
<code>__pred</code>	A predicate.

Returns

A pair designating the ends of the resulting sequences.

Copies each element in the range `[__first, __last)` for which `__pred` returns true to the range beginning at `out_true` and each element for which `__pred` returns false to `__out_false`.

Definition at line 825 of file `stl_algo.h`.

References `std::__find_if()`.

```
3.51.2.15 template<typename _ForwardIterator, typename _Predicate> _ForwardIterator std::partition_point ( _ForwardIterator
    __first, _ForwardIterator __last, _Predicate __pred )
```

Find the partition point of a partitioned range.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__pred</code>	A predicate.

Returns

An iterator `mid` such that `all_of(__first, mid, __pred)` and `none_of(mid, __last, __pred)` are both true.

Definition at line 600 of file `stl_algo.h`.

References `std::advance()`, and `std::distance()`.

3.51.2.16 `template<typename _RandomAccessIterator, typename _RandomNumberGenerator> void std::random_shuffle (`
`_RandomAccessIterator __first, _RandomAccessIterator __last, _RandomNumberGenerator && __rand)`

Shuffle the elements of a sequence using a random number generator.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__rand</code>	The RNG functor or function.

Returns

Nothing.

Reorders the elements in the range `[__first,__last)` using `__rand` to provide a random distribution. Calling `__rand(↔ N)` for a positive integer `N` should return a randomly chosen integer from the range `[0,N)`.

Definition at line 4474 of file `stl_algo.h`.

References `std::iter_swap()`.

3.51.2.17 `template<typename _ForwardIterator, typename _Tp> _ForwardIterator std::remove (_ForwardIterator __first,`
`_ForwardIterator __last, const _Tp & __value) [inline]`

Remove elements from a sequence.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__value</code>	The value to be removed.

Returns

An iterator designating the end of the resulting sequence.

All elements equal to `__value` are removed from the range `[__first,__last)`.

`remove()` is stable, so the relative order of elements that are not removed is unchanged.

Elements between the end of the resulting sequence and `__last` are still present, but their value is unspecified.

Definition at line 893 of file `stl_algo.h`.

3.51.2.18 `template<typename _InputIterator , typename _OutputIterator , typename _Tp > _OutputIterator std::remove_copy (`
`_InputIterator __first, _InputIterator __last, _OutputIterator __result, const _Tp & __value) [inline]`

Copy a sequence, removing elements of a given value.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.
<code>__value</code>	The value to be removed.

Returns

An iterator designating the end of the resulting sequence.

Copies each element in the range `[__first,__last)` not equal to `__value` to the range beginning at `__result`. `remove_copy()` is stable, so the relative order of elements that are copied is unchanged.

Definition at line 667 of file `stl_algo.h`.

3.51.2.19 `template<typename _InputIterator , typename _OutputIterator , typename _Predicate > _OutputIterator`
`std::remove_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Predicate __pred)`
`[inline]`

Copy a sequence, removing elements for which a predicate is true.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.
<code>__pred</code>	A predicate.

Returns

An iterator designating the end of the resulting sequence.

Copies each element in the range `[__first,__last)` for which `__pred` returns false to the range beginning at `__result`. `remove_copy_if()` is stable, so the relative order of elements that are copied is unchanged.

Definition at line 700 of file `stl_algo.h`.

3.51.2.20 `template<typename _ForwardIterator , typename _Predicate > _ForwardIterator std::remove_if (_ForwardIterator __first,`
`_ForwardIterator __last, _Predicate __pred) [inline]`

Remove elements from a sequence using a predicate.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__pred</code>	A predicate.

Returns

An iterator designating the end of the resulting sequence.

All elements for which `__pred` returns true are removed from the range `[__first,__last)`.

`remove_if()` is stable, so the relative order of elements that are not removed is unchanged.

Elements between the end of the resulting sequence and `__last` are still present, but their value is unspecified.

Definition at line 926 of file `stl_algo.h`.

3.51.2.21 `template<typename _ForwardIterator, typename _Tp> void std::replace (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __old_value, const _Tp & __new_value)`

Replace each occurrence of one value in a sequence with another value.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__old_value</code>	The value to be replaced.
<code>__new_value</code>	The replacement value.

Returns

`replace()` returns no value.

For each iterator `i` in the range `[__first,__last)` if `*i == __old_value` then the assignment `*i = __new_value` is performed.

Definition at line 4235 of file `stl_algo.h`.

3.51.2.22 `template<typename _InputIterator, typename _OutputIterator, typename _Predicate, typename _Tp> _OutputIterator std::replace_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Predicate __pred, const _Tp & __new_value) [inline]`

Copy a sequence, replacing each value for which a predicate returns true with another value.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.
<code>__pred</code>	A predicate.
<code>__new_value</code>	The replacement value.

Returns

The end of the output sequence, `__result+(__last-__first)`.

Copies each element in the range `[__first,__last)` to the range `[__result,__result+(__last-__first))` replacing elements for which `__pred` returns true with `__new_value`.

Definition at line 3168 of file `stl_algo.h`.

```
3.51.2.23  template<typename _ForwardIterator, typename _Predicate, typename _Tp> void std::replace_if ( _ForwardIterator
            __first, _ForwardIterator __last, _Predicate __pred, const _Tp & __new_value )
```

Replace each value in a sequence for which a predicate returns true with another value.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__pred</code>	A predicate.
<code>__new_value</code>	The replacement value.

Returns

`replace_if()` returns no value.

For each iterator `i` in the range `[__first,__last)` if `__pred(*i)` is true then the assignment `*i = __new_value` is performed.

Definition at line 4267 of file `stl_algo.h`.

```
3.51.2.24  template<typename _BidirectionalIterator> void std::reverse ( _BidirectionalIterator __first, _BidirectionalIterator __last
            ) [inline]
```

Reverse a sequence.

Parameters

<code>__first</code>	A bidirectional iterator.
<code>__last</code>	A bidirectional iterator.

Returns

`reverse()` returns no value.

Reverses the order of the elements in the range `[__first, __last)`, so that the first element becomes the last etc. For every `i` such that $0 \leq i < (\text{__last} - \text{__first})/2$, `reverse()` swaps `*(__first+i)` and `*(__last-(i+1))`

Definition at line 1177 of file `stl_algo.h`.

References `std::__iterator_category()`, and `std::__reverse()`.

3.51.2.25 `template<typename _BidirectionalIterator, typename _OutputIterator> _OutputIterator std::reverse_copy (`
`_BidirectionalIterator __first, _BidirectionalIterator __last, _OutputIterator __result)`

Copy a sequence, reversing its elements.

Parameters

<code>__first</code>	A bidirectional iterator.
<code>__last</code>	A bidirectional iterator.
<code>__result</code>	An output iterator.

Returns

An iterator designating the end of the resulting sequence.

Copies the elements in the range `[__first, __last)` to the range `[__result, __result+(__last-__first))` such that the order of the elements is reversed. For every `i` such that $0 \leq i < (\text{__last} - \text{__first})$, `reverse_copy()` performs the assignment `*(__result+(__last-__first)-1-i) = *(__first+i)`. The ranges `[__first, __last)` and `[__result, __result+(__last-__first))` must not overlap.

Definition at line 1204 of file `stl_algo.h`.

3.51.2.26 `template<typename _ForwardIterator> _ForwardIterator std::V2::rotate (_ForwardIterator __first, _ForwardIterator`
`__middle, _ForwardIterator __last) [inline]`

Rotate the elements of a sequence.

Parameters

<code>__first</code>	A forward iterator.
<code>__middle</code>	A forward iterator.
<code>__last</code>	A forward iterator.

Returns

`first + (last - middle)`.

Rotates the elements of the range `[__first,__last)` by `(__middle - __first)` positions so that the element at `__middle` is moved to `__first`, the element at `__middle+1` is moved to `__first+1` and so on for each element in the range `[__first,__last)`.

This effectively swaps the ranges `[__first,__middle)` and `[__middle,__last)`.

Performs `*(__first+(n+(__last-__middle))%(__last-__first))=*(__first+n)` for each `n` in the range `[0,__last-__first)`.

Definition at line 1431 of file `stl_algo.h`.

References `std::__iterator_category()`.

Referenced by `std::__merge_without_buffer()`, `std::__rotate_adaptive()`, and `std::__stable_partition_adaptive()`.

3.51.2.27 `template<typename _ForwardIterator, typename _OutputIterator> _OutputIterator std::rotate_copy (_ForwardIterator __first, _ForwardIterator __middle, _ForwardIterator __last, _OutputIterator __result) [inline]`

Copy a sequence, rotating its elements.

Parameters

<code>__first</code>	A forward iterator.
<code>__middle</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__result</code>	An output iterator.

Returns

An iterator designating the end of the resulting sequence.

Copies the elements of the range `[__first,__last)` to the range beginning at

Returns

, rotating the copied elements by `(__middle-__first)` positions so that the element at `__middle` is moved to `__result`, the element at `__middle+1` is moved to `__result+1` and so on for each element in the range `[__first,__last)`.

Performs `*(__result+(n+(__last-__middle))%(__last-__first))=*(__first+n)` for each `n` in the range `[0,__last-__first)`.

Definition at line 1468 of file `stl_algo.h`.

3.51.2.28 `template<typename _RandomAccessIterator, typename _UniformRandomNumberGenerator> void std::shuffle (_RandomAccessIterator __first, _RandomAccessIterator __last, _UniformRandomNumberGenerator && __g)`

Shuffle the elements of a sequence using a uniform random number generator.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__g</code>	A UniformRandomNumberGenerator (26.5.1.3).

Returns

Nothing.

Reorders the elements in the range `[__first,__last)` using `__g` to provide random numbers.

Definition at line 3719 of file `stl_algo.h`.

References `std::iter_swap()`.

3.51.2.29 `template<typename _ForwardIterator, typename _Predicate> _ForwardIterator std::stable_partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred) [inline]`

Move elements for which a predicate is true to the beginning of a sequence, preserving relative ordering.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__pred</code>	A predicate functor.

Returns

An iterator `middle` such that `__pred(i)` is true for each iterator `i` in the range `[first,middle)` and false for each `i` in the range `[middle,last)`.

Performs the same function as `partition()` with the additional guarantee that the relative ordering of elements in each group is preserved, so any two elements `x` and `y` in the range `[__first,__last)` such that `__pred(x) == __pred(y)` will have the same relative ordering after calling `stable_partition()`.

Definition at line 1648 of file `stl_algo.h`.

3.51.2.30 `template<typename _ForwardIterator1, typename _ForwardIterator2> _ForwardIterator2 std::swap_ranges (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2)`

Swap the elements of two sequences.

Parameters

<code>__first1</code>	A forward iterator.
<code>__last1</code>	A forward iterator.
<code>__first2</code>	A forward iterator.

Returns

An iterator equal to `first2+(last1-first1)`.

Swaps each element in the range `[first1,last1)` with the corresponding element in the range `[first2,(last1-first1))`. The ranges must not overlap.

Definition at line 166 of file `stl_algobase.h`.

References `std::iter_swap()`.

Referenced by `std::_V2::__rotate()`.

3.51.2.31 `template<typename _InputIterator , typename _OutputIterator , typename _UnaryOperation > _OutputIterator
std::transform (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _UnaryOperation __unary_op)`

Perform an operation on a sequence.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.
<code>__unary_op</code>	A unary operator.

Returns

An output iterator equal to `__result+(__last-__first)`.

Applies the operator to each element in the input range and assigns the results to successive elements of the output sequence. Evaluates `*(__result+N)=unary_op(*(__first+N))` for each `N` in the range `[0,__last-__first)`.

`unary_op` must not alter its argument.

Definition at line 4166 of file `stl_algo.h`.

3.51.2.32 `template<typename _InputIterator1 , typename _InputIterator2 , typename _OutputIterator , typename _BinaryOperation >
_OutputIterator std::transform (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _OutputIterator
__result, _BinaryOperation __binary_op)`

Perform an operation on corresponding elements of two sequences.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.
<code>__result</code>	An output iterator.
<code>__binary_op</code>	A binary operator.

Returns

An output iterator equal to `result+(last-first)`.

Applies the operator to the corresponding elements in the two input ranges and assigns the results to successive elements of the output sequence. Evaluates `*(__result+N)=__binary_op(*(__first1+N),*(__first2+N))` for each `N` in the range `[0,__last1-__first1)`.

`binary_op` must not alter either of its arguments.

Definition at line 4203 of file `stl_algo.h`.

3.51.2.33 `template<typename _ForwardIterator > _ForwardIterator std::unique (_ForwardIterator __first, _ForwardIterator __last)`
`[inline]`

Remove consecutive duplicate values from a sequence.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.

Returns

An iterator designating the end of the resulting sequence.

Removes all but the first element from each group of consecutive values that compare equal. `unique()` is stable, so the relative order of elements that are not removed is unchanged. Elements between the end of the resulting sequence and `__last` are still present, but their value is unspecified.

Definition at line 992 of file `stl_algo.h`.

3.51.2.34 `template<typename _ForwardIterator, typename _BinaryPredicate > _ForwardIterator std::unique (_ForwardIterator __first, _ForwardIterator __last, _BinaryPredicate __binary_pred)` `[inline]`

Remove consecutive values from a sequence using a predicate.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__binary_pred</code>	A binary predicate.

Returns

An iterator designating the end of the resulting sequence.

Removes all but the first element from each group of consecutive values for which `__binary_pred` returns true.

`unique()` is stable, so the relative order of elements that are not removed is unchanged. Elements between the end of the resulting sequence and `__last` are still present, but their value is unspecified.

Definition at line 1022 of file `stl_algo.h`.

3.51.2.35 `template<typename _InputIterator, typename _OutputIterator> _OutputIterator std::unique_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result) [inline]`

Copy a sequence, removing consecutive duplicate values.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.

Returns

An iterator designating the end of the resulting sequence.

Copies each element in the range `[__first, __last)` to the range beginning at `__result`, except that only the first element is copied from groups of consecutive elements that compare equal. `unique_copy()` is stable, so the relative order of elements that are copied is unchanged.

`_GLIBCXX_RESOLVE_LIB_DEFECTS` DR 241. Does `unique_copy()` require `CopyConstructible` and `Assignable`?

`_GLIBCXX_RESOLVE_LIB_DEFECTS` DR 538. 241 again: Does `unique_copy()` require `CopyConstructible` and `Assignable`?

Definition at line 4366 of file `stl_algo.h`.

References `std::__iterator_category()`, and `std::__unique_copy()`.

3.51.2.36 `template<typename _InputIterator, typename _OutputIterator, typename _BinaryPredicate> _OutputIterator std::unique_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _BinaryPredicate __binary_pred) [inline]`

Copy a sequence, removing consecutive values using a predicate.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.
<code>__binary_pred</code>	A binary predicate.

Returns

An iterator designating the end of the resulting sequence.

Copies each element in the range `[__first, __last)` to the range beginning at `__result`, except that only the first element is copied from groups of consecutive elements for which `__binary_pred` returns true. `unique_copy()` is stable, so the relative order of elements that are copied is unchanged.

`_GLIBCXX_RESOLVE_LIB_DEFECTS` DR 241. Does `unique_copy()` require `CopyConstructible` and `Assignable`?

Definition at line 4407 of file `stl_algo.h`.

References `std::__iterator_category()`, `std::__unique_copy()`, and `std::iter_swap()`.

3.52 Mutexes

Collaboration diagram for Mutexes:



Classes

- struct [std::adopt_lock_t](#)
- struct [std::defer_lock_t](#)
- class [std::lock_guard< _Mutex >](#)
- class [std::mutex](#)
- struct [std::try_to_lock_t](#)
- class [std::unique_lock< _Mutex >](#)

Functions

- template<typename [_Mutex](#) >
void [std::swap](#) ([unique_lock< _Mutex >](#) &__x, [unique_lock< _Mutex >](#) &__y) noexcept

Variables

- constexpr [adopt_lock_t](#) [std::adopt_lock](#)
- constexpr [defer_lock_t](#) [std::defer_lock](#)
- constexpr [try_to_lock_t](#) [std::try_to_lock](#)
- template<typename [_Mutex](#) >
void [std::swap](#) ([shared_lock< _Mutex >](#) &__x, [shared_lock< _Mutex >](#) &__y) noexcept
- #define [__cpp_lib_shared_timed_mutex](#)
- using [std::__shared_timed_mutex_base](#) = [__shared_mutex_cv](#)

3.52.1 Detailed Description

Classes for mutex support.

3.52.2 Macro Definition Documentation

3.52.2.1 `#define __cpp_lib_shared_timed_mutex`

Swap specialization for `shared_lock`.

Definition at line 59 of file `shared_mutex`.

3.52.3 Typedef Documentation

3.52.3.1 `using std::__shared_timed_mutex_base = typedef __shared_mutex_cv`

Swap specialization for `shared_lock`.

Definition at line 357 of file `shared_mutex`.

3.52.4 Function Documentation

3.52.4.1 `template<typename _Mutex > void std::swap (unique_lock< _Mutex > & __x, unique_lock< _Mutex > & __y)` `[inline], [noexcept]`

Swap overload for `unique_lock` objects.

Definition at line 363 of file `std_mutex.h`.

3.52.4.2 `template<typename _Mutex > void std::swap (shared_lock< _Mutex > & __x, shared_lock< _Mutex > & __y)` `[noexcept]`

Swap specialization for `shared_lock`.

Definition at line 675 of file `shared_mutex`.

3.52.5 Variable Documentation

3.52.5.1 `constexpr adopt_lock_t std::adopt_lock`

Tag used to make a scoped lock take ownership of a locked mutex.

Definition at line 148 of file `std_mutex.h`.

3.52.5.2 `constexpr defer_lock_t std::defer_lock`

Tag used to prevent a scoped lock from acquiring ownership of a mutex.

Definition at line 142 of file `std_mutex.h`.

3.52.5.3 `constexpr try_to_lock_t std::try_to_lock`

Tag used to prevent a scoped lock from blocking if a mutex is locked.

Definition at line 145 of file `std_mutex.h`.

3.53 Negators

Collaboration diagram for Negators:



Classes

- class `std::binary_negate<_Predicate>`
- class `std::unary_negate<_Predicate>`

Functions

- `template<typename _Predicate>`
`_GLIBCXX14_CONSTEXPR unary_negate<_Predicate> std::not1 (const _Predicate &__pred)`
- `template<typename _Predicate>`
`_GLIBCXX14_CONSTEXPR binary_negate<_Predicate> std::not2 (const _Predicate &__pred)`

3.53.1 Detailed Description

The functions `not1` and `not2` each take a predicate functor and return an instance of `unary_negate` or `binary_negate`, respectively. These classes are functors whose `operator()` performs the stored predicate function and then returns the negation of the result.

For example, given a vector of integers and a trivial predicate,

```

struct IntGreaterThanThree
: public std::unary_function<int, bool>
{
    bool operator() (int x) { return x > 3; }
};

std::find_if (v.begin(), v.end(), not1(IntGreaterThanThree()));
  
```

The call to `find_if` will locate the first index (*i*) of *v* for which `!(v[i] > 3)` is true.

The `not1/unary_negate` combination works on predicates taking a single argument. The `not2/binary_negate` combination works on predicates which take two arguments.

3.53.2 Function Documentation

3.53.2.1 `template<typename _Predicate > _GLIBCXX14_CONSTEXPR unary_negate<_Predicate> std::not1 (const _Predicate & __pred) [inline]`

One of the [negation functors](#).

Definition at line 762 of file `stl_function.h`.

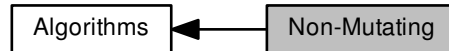
3.53.2.2 `template<typename _Predicate > _GLIBCXX14_CONSTEXPR binary_negate<_Predicate> std::not2 (const _Predicate & __pred) [inline]`

One of the [negation functors](#).

Definition at line 790 of file `stl_function.h`.

3.54 Non-Mutating

Collaboration diagram for Non-Mutating:



Functions

- `template<typename _ForwardIterator >`
`_ForwardIterator std::adjacent_find (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _BinaryPredicate >`
`_ForwardIterator std::adjacent_find (_ForwardIterator __first, _ForwardIterator __last, _BinaryPredicate __↔
binary_pred)`
- `template<typename _InputIterator, typename _Predicate >`
`bool std::all_of (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _Predicate >`
`bool std::any_of (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _Tp >`
`iterator_traits< _InputIterator >::difference_type std::count (_InputIterator __first, _InputIterator __last, const _Tp
&__value)`
- `template<typename _InputIterator, typename _Predicate >`
`iterator_traits< _InputIterator >::difference_type std::count_if (_InputIterator __first, _InputIterator __last, _↔
Predicate __pred)`
- `template<typename _Iter1, typename _Iter2, typename _BinaryPredicate >`
`bool std::equal (_Iter1 __first1, _Iter1 __last1, _Iter2 __first2, _BinaryPredicate __binary_pred)`
- `template<typename _II1, typename _II2 >`
`bool std::equal (_II1 __first1, _II1 __last1, _II2 __first2)`
- `template<typename _II1, typename _II2 >`
`bool std::equal (_II1 __first1, _II1 __last1, _II2 __first2, _II2 __last2)`
- `template<typename _Iter1, typename _Iter2, typename _BinaryPredicate >`
`bool std::equal (_Iter1 __first1, _Iter1 __last1, _Iter2 __first2, _Iter2 __last2, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator, typename _Tp >`
`_InputIterator std::find (_InputIterator __first, _InputIterator __last, const _Tp &__val)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`_ForwardIterator1 std::find_end (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __↔
first2, _ForwardIterator2 __last2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`_ForwardIterator1 std::find_end (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __↔
first2, _ForwardIterator2 __last2, _BinaryPredicate __comp)`
- `template<typename _InputIterator, typename _ForwardIterator >`
`_InputIterator std::find_first_of (_InputIterator __first1, _InputIterator __last1, _ForwardIterator __first2, _↔
ForwardIterator __last2)`

- `template<typename _InputIterator, typename _ForwardIterator, typename _BinaryPredicate >`
`_InputIterator std::find_first_of (_InputIterator __first1, _InputIterator __last1, _ForwardIterator __first2, ↵`
`_ForwardIterator __last2, _BinaryPredicate __comp)`
- `template<typename _InputIterator, typename _Predicate >`
`_InputIterator std::find_if (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _Predicate >`
`_InputIterator std::find_if_not (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _Function >`
`_Function std::for_each (_InputIterator __first, _InputIterator __last, _Function __f)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, ↵`
`_BinaryPredicate __pred)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, ↵`
`_ForwardIterator2 __last2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, ↵`
`_ForwardIterator2 __last2, _BinaryPredicate __pred)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`pair< _InputIterator1, _InputIterator2 > std::mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _Input↵`
`Iterator2 __first2)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`pair< _InputIterator1, _InputIterator2 > std::mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _Input↵`
`Iterator2 __first2, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`pair< _InputIterator1, _InputIterator2 > std::mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _Input↵`
`Iterator2 __first2, _InputIterator2 __last2)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`pair< _InputIterator1, _InputIterator2 > std::mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _Input↵`
`Iterator2 __first2, _InputIterator2 __last2, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator, typename _Predicate >`
`bool std::none_of (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`_ForwardIterator1 std::search (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2,`
`_ForwardIterator2 __last2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`_ForwardIterator1 std::search (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2,`
`_ForwardIterator2 __last2, _BinaryPredicate __predicate)`
- `template<typename _ForwardIterator, typename _Integer, typename _Tp >`
`_ForwardIterator std::search_n (_ForwardIterator __first, _ForwardIterator __last, _Integer __count, const _Tp`
`&__val)`
- `template<typename _ForwardIterator, typename _Integer, typename _Tp, typename _BinaryPredicate >`
`_ForwardIterator std::search_n (_ForwardIterator __first, _ForwardIterator __last, _Integer __count, const _Tp`
`&__val, _BinaryPredicate __binary_pred)`

3.54.1 Detailed Description

3.54.2 Function Documentation

3.54.2.1 `template<typename _ForwardIterator > _ForwardIterator std::adjacent_find (_ForwardIterator __first, _ForwardIterator __last) [inline]`

Find two adjacent values in a sequence that are equal.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.

Returns

The first iterator `i` such that `i` and `i+1` are both valid iterators in `[__first,__last)` and such that `*i == *(i+1)`, or `__last` if no such iterator exists.

Definition at line 3911 of file `stl_algo.h`.

3.54.2.2 `template<typename _ForwardIterator , typename _BinaryPredicate > _ForwardIterator std::adjacent_find (_ForwardIterator __first, _ForwardIterator __last, _BinaryPredicate __binary_pred) [inline]`

Find two adjacent values in a sequence using a predicate.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__binary_pred</code>	A binary predicate.

Returns

The first iterator `i` such that `i` and `i+1` are both valid iterators in `[__first,__last)` and such that `__binary_pred(*i,*(i+1))` is true, or `__last` if no such iterator exists.

Definition at line 3936 of file `stl_algo.h`.

3.54.2.3 `template<typename _InputIterator , typename _Predicate > bool std::all_of (_InputIterator __first, _InputIterator __last, _Predicate __pred) [inline]`

Checks that a predicate is true for all the elements of a sequence.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__pred</code>	A predicate.

Returns

True if the check is true, false otherwise.

Returns true if `__pred` is true for each element in the range `[__first,__last)`, and false otherwise.

Definition at line 508 of file `stl_algo.h`.

References `std::find_if_not()`.

```
3.54.2.4  template<typename _InputIterator , typename _Predicate > bool std::any_of ( _InputIterator __first, _InputIterator __last,
    _Predicate __pred ) [inline]
```

Checks that a predicate is false for at least an element of a sequence.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__pred</code>	A predicate.

Returns

True if the check is true, false otherwise.

Returns true if an element exists in the range `[__first,__last)` such that `__pred` is true, and false otherwise.

Definition at line 543 of file `stl_algo.h`.

References `std::none_of()`.

```
3.54.2.5  template<typename _InputIterator , typename _Tp > iterator_traits<_InputIterator>::difference_type std::count (
    _InputIterator __first, _InputIterator __last, const _Tp & __value ) [inline]
```

Count the number of copies of a value in a sequence.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__value</code>	The value to be counted.

Returns

The number of iterators `i` in the range `[__first,__last)` for which `*i == __value`

Definition at line 3961 of file `stl_algo.h`.

3.54.2.6 `template<typename _InputIterator, typename _Predicate> iterator_traits<_InputIterator>::difference_type std::count_if (_InputIterator __first, _InputIterator __last, _Predicate __pred) [inline]`

Count the elements of a sequence for which a predicate is true.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__pred</code>	A predicate.

Returns

The number of iterators `i` in the range `[__first, __last)` for which `__pred(*i)` is true.

Definition at line 3984 of file `stl_algo.h`.

3.54.2.7 `template<typename _Iter1, typename _Iter2, typename _BinaryPredicate> bool std::equal (_Iter1 __first1, _Iter1 __last1, _Iter2 __first2, _BinaryPredicate __binary_pred) [inline]`

Tests a range for element-wise equality.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.
<code>__binary_pred</code>	A binary predicate functor .

Returns

A boolean true or false.

This compares the elements of two ranges using the `binary_pred` parameter, and returns true or false depending on whether all of the corresponding elements of the ranges are equal.

Definition at line 1073 of file `stl_algobase.h`.

3.54.2.8 `template<typename _II1, typename _II2> bool std::equal (_II1 __first1, _II1 __last1, _II2 __first2) [inline]`

Tests a range for element-wise equality.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.

Returns

A boolean true or false.

This compares the elements of two ranges using `==` and returns true or false depending on whether all of the corresponding elements of the ranges are equal.

Definition at line 1041 of file `stl_algobase.h`.

```
3.54.2.9  template<typename _I1, typename _I2> bool std::equal ( _I1 __first1, _I1 __last1, _I2 __first2, _I2 __last2 )
           [inline]
```

Tests a range for element-wise equality.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.
<code>__last2</code>	An input iterator.

Returns

A boolean true or false.

This compares the elements of two ranges using `==` and returns true or false depending on whether all of the corresponding elements of the ranges are equal.

Definition at line 1106 of file `stl_algobase.h`.

References `std::distance()`, and `std::equal()`.

```
3.54.2.10 template<typename _Iiter1, typename _Iiter2, typename _BinaryPredicate> bool std::equal ( _Iiter1 __first1, _Iiter1
__last1, _Iiter2 __first2, _Iiter2 __last2, _BinaryPredicate __binary_pred ) [inline]
```

Tests a range for element-wise equality.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.
<code>__last2</code>	An input iterator.
<code>__binary_pred</code>	A binary predicate functor .

Returns

A boolean true or false.

This compares the elements of two ranges using the binary_pred parameter, and returns true or false depending on whether all of the corresponding elements of the ranges are equal.

Definition at line 1155 of file stl_algobase.h.

References std::distance().

Referenced by std::equal(), std::fill_n(), and std::operator==().

3.54.2.11 `template<typename _InputIterator, typename _Tp> _InputIterator std::find (_InputIterator __first, _InputIterator __last, const _Tp &__val) [inline]`

Find the first occurrence of a value in a sequence.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__val</code>	The value to find.

Returns

The first iterator `i` in the range `[__first,__last)` such that `*i == __val`, or `__last` if no such iterator exists.

Definition at line 3784 of file stl_algo.h.

References std::__find_if().

3.54.2.12 `template<typename _ForwardIterator1, typename _ForwardIterator2> _ForwardIterator1 std::find_end (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2 __last2) [inline]`

Find last matching subsequence in a sequence.

Parameters

<code>__first1</code>	Start of range to search.
<code>__last1</code>	End of range to search.
<code>__first2</code>	Start of sequence to match.
<code>__last2</code>	End of sequence to match.

Returns

The last iterator `i` in the range `[__first1, __last1 - (__last2 - __first2))` such that `*(i+N) == *(__first2+N)` for each `N` in the range `[0, __last2 - __first2)`, or `__last1` if no such iterator exists.

Searches the range `[__first1, __last1)` for a sub-sequence that compares equal value-by-value with the sequence given by `[__first2, __last2)` and returns an iterator to the `__first` element of the sub-sequence, or `__last1` if the sub-sequence is not found. The sub-sequence will be the last such subsequence contained in `[__first1, __last1)`.

Because the sub-sequence must lie completely within the range `[__first1, __last1)` it must start at a position less than `__last1 - (__last2 - __first2)` where `__last2 - __first2` is the length of the sub-sequence. This means that the returned iterator `i` will be in the range `[__first1, __last1 - (__last2 - __first2))`.

Definition at line 425 of file `stl_algo.h`.

References `std::__iterator_category()`.

```
3.54.2.13 template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate > _ForwardIterator1
std::find_end ( _ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2
__last2, _BinaryPredicate __comp ) [inline]
```

Find last matching subsequence in a sequence using a predicate.

Parameters

<code>__first1</code>	Start of range to search.
<code>__last1</code>	End of range to search.
<code>__first2</code>	Start of sequence to match.
<code>__last2</code>	End of sequence to match.
<code>__comp</code>	The predicate to use.

Returns

The last iterator `i` in the range `[__first1, __last1 - (__last2 - __first2))` such that `predicate(*(i+N), (__first2+N))` is true for each `N` in the range `[0, __last2 - __first2)`, or `__last1` if no such iterator exists.

Searches the range `[__first1, __last1)` for a sub-sequence that compares equal value-by-value with the sequence given by `[__first2, __last2)` using `comp` as a predicate and returns an iterator to the first element of the sub-sequence, or `__last1` if the sub-sequence is not found. The sub-sequence will be the last such subsequence contained in `[__first1, __last1)`.

Because the sub-sequence must lie completely within the range `[__first1, __last1)` it must start at a position less than `__last1 - (__last2 - __first2)` where `__last2 - __first2` is the length of the sub-sequence. This means that the returned iterator `i` will be in the range `[__first1, __last1 - (__last2 - __first2))`.

Definition at line 474 of file `stl_algo.h`.

References `std::__iterator_category()`.

```
3.54.2.14 template<typename _InputIterator, typename _ForwardIterator > _InputIterator std::find_first_of ( _InputIterator __first1,
_InputIterator __last1, _ForwardIterator __first2, _ForwardIterator __last2 )
```

Find element from a set in a sequence.

Parameters

<code>__first1</code>	Start of range to search.
<code>__last1</code>	End of range to search.
<code>__first2</code>	Start of match candidates.
<code>__last2</code>	End of match candidates.

Returns

The first iterator `i` in the range `[__first1, __last1)` such that `*i == *(i2)` such that `i2` is an iterator in `[__first2, __last2)`, or `__last1` if no such iterator exists.

Searches the range `[__first1, __last1)` for an element that is equal to some element in the range `[__first2, __last2)`. If found, returns an iterator in the range `[__first1, __last1)`, otherwise returns `__last1`.

Definition at line 3839 of file `stl_algo.h`.

```
3.54.2.15  template<typename _InputIterator, typename _ForwardIterator, typename _BinaryPredicate> _InputIterator
            std::find_first_of ( _InputIterator __first1, _InputIterator __last1, _ForwardIterator __first2, _ForwardIterator __last2,
                               _BinaryPredicate __comp )
```

Find element from a set in a sequence using a predicate.

Parameters

<code>__first1</code>	Start of range to search.
<code>__last1</code>	End of range to search.
<code>__first2</code>	Start of match candidates.
<code>__last2</code>	End of match candidates.
<code>__comp</code>	Predicate to use.

Returns

The first iterator `i` in the range `[__first1, __last1)` such that `comp(*i, *(i2))` is true and `i2` is an iterator in `[__first2, __last2)`, or `__last1` if no such iterator exists.

Searches the range `[__first1, __last1)` for an element that is equal to some element in the range `[__first2, __last2)`. If found, returns an iterator in the range `[__first1, __last1)`, otherwise returns `__last1`.

Definition at line 3880 of file `stl_algo.h`.

```
3.54.2.16  template<typename _InputIterator, typename _Predicate> _InputIterator std::find_if ( _InputIterator __first,
                                                _InputIterator __last, _Predicate __pred ) [inline]
```

Find the first element in a sequence for which a predicate is true.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__pred</code>	A predicate.

Returns

The first iterator `i` in the range `[__first,__last)` such that `__pred(*i)` is true, or `__last` if no such iterator exists.

Definition at line 3808 of file `stl_algo.h`.

References `std::__find_if()`.

Referenced by `std::none_of()`.

```
3.54.2.17 template<typename _InputIterator, typename _Predicate> _InputIterator std::find_if_not ( _InputIterator __first,
    _InputIterator __last, _Predicate __pred ) [inline]
```

Find the first element in a sequence for which a predicate is false.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__pred</code>	A predicate.

Returns

The first iterator `i` in the range `[__first,__last)` such that `__pred(*i)` is false, or `__last` if no such iterator exists.

Definition at line 558 of file `stl_algo.h`.

References `std::__find_if_not()`.

Referenced by `std::all_of()`, and `std::is_partitioned()`.

```
3.54.2.18 template<typename _InputIterator, typename _Function> _Function std::for_each ( _InputIterator __first, _InputIterator
    __last, _Function __f )
```

Apply a function to every element of a sequence.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__f</code>	A unary function object.

Returns

`__f` (`std::move(__f)` in C++0x).

Applies the function object `__f` to each element in the range `[first,last)`. `__f` must not modify the order of the sequence. If `__f` has a return value it is ignored.

Definition at line 3763 of file `stl_algo.h`.

3.54.2.19 `template<typename _ForwardIterator1, typename _ForwardIterator2 > bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2) [inline]`

Checks whether a permutation of the second sequence is equal to the first sequence.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.

Returns

true if there exists a permutation of the elements in the range `[__first2, __first2 + (__last1 - __first1))`, beginning with `ForwardIterator2` begin, such that `equal(__first1, __last1, begin)` returns true; otherwise, returns false.

Definition at line 3539 of file `stl_algo.h`.

3.54.2.20 `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate > bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _BinaryPredicate __pred) [inline]`

Checks whether a permutation of the second sequence is equal to the first sequence.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__pred</code>	A binary predicate.

Returns

true if there exists a permutation of the elements in the range `[__first2, __first2 + (__last1 - __first1))`, beginning with `ForwardIterator2` begin, such that `equal(__first1, __last1, __begin, __pred)` returns true; otherwise, returns false.

Definition at line 3571 of file `stl_algo.h`.

References `std::__find_if()`, and `std::distance()`.

3.54.2.21 `template<typename _ForwardIterator1, typename _ForwardIterator2> bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2 __last2) [inline]`

Checks whether a permutaion of the second sequence is equal to the first sequence.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__last2</code>	End of first range.

Returns

true if there exists a permutation of the elements in the range `[__first2, __last2)`, beginning with `ForwardIterator2` begin, such that `equal(__first1, __last1, begin)` returns true; otherwise, returns false.

Definition at line 3663 of file `stl_algo.h`.

3.54.2.22 `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate> bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2 __last2, _BinaryPredicate __pred) [inline]`

Checks whether a permutation of the second sequence is equal to the first sequence.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__last2</code>	End of first range.
<code>__pred</code>	A binary predicate.

Returns

true if there exists a permutation of the elements in the range `[__first2, __last2)`, beginning with `ForwardIterator2` begin, such that `equal(__first1, __last1, __begin, __pred)` returns true; otherwise, returns false.

Definition at line 3691 of file `stl_algo.h`.

3.54.2.23 `template<typename _InputIterator1, typename _InputIterator2> pair<_InputIterator1, _InputIterator2> std::mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2) [inline]`

Finds the places in ranges which don't match.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.

Returns

A pair of iterators pointing to the first mismatch.

This compares the elements of two ranges using `==` and returns a pair of iterators. The first iterator points into the first range, the second iterator points into the second range, and the elements pointed to by the iterators are not equal.

Definition at line 1282 of file `stl_algobase.h`.

```
3.54.2.24  template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate > pair<_InputIterator1,
            _InputIterator2> std::mismatch ( _InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2,
            _BinaryPredicate __binary_pred ) [inline]
```

Finds the places in ranges which don't match.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.
<code>__binary_pred</code>	A binary predicate functor .

Returns

A pair of iterators pointing to the first mismatch.

This compares the elements of two ranges using the `binary_pred` parameter, and returns a pair of iterators. The first iterator points into the first range, the second iterator points into the second range, and the elements pointed to by the iterators are not equal.

Definition at line 1316 of file `stl_algobase.h`.

```
3.54.2.25  template<typename _InputIterator1, typename _InputIterator2 > pair<_InputIterator1, _InputIterator2> std::mismatch (
            _InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2 ) [inline]
```

Finds the places in ranges which don't match.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.
<code>__last2</code>	An input iterator.

Returns

A pair of iterators pointing to the first mismatch.

This compares the elements of two ranges using `==` and returns a pair of iterators. The first iterator points into the first range, the second iterator points into the second range, and the elements pointed to by the iterators are not equal.

Definition at line 1362 of file `stl_algobase.h`.

```
3.54.2.26  template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate > pair<_InputIterator1,
            _InputIterator2> std::mismatch ( _InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2
            __last2, _BinaryPredicate __binary_pred ) [inline]
```

Finds the places in ranges which don't match.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.
<code>__last2</code>	An input iterator.
<code>__binary_pred</code>	A binary predicate functor .

Returns

A pair of iterators pointing to the first mismatch.

This compares the elements of two ranges using the `binary_pred` parameter, and returns a pair of iterators. The first iterator points into the first range, the second iterator points into the second range, and the elements pointed to by the iterators are not equal.

Definition at line 1398 of file `stl_algobase.h`.

```
3.54.2.27  template<typename _InputIterator, typename _Predicate > bool std::none_of ( _InputIterator __first, _InputIterator
            __last, _Predicate __pred ) [inline]
```

Checks that a predicate is false for all the elements of a sequence.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__pred</code>	A predicate.

Returns

True if the check is true, false otherwise.

Returns true if `__pred` is false for each element in the range `[__first,__last)`, and false otherwise.

Definition at line 525 of file `stl_algo.h`.

References `std::find_if()`.

Referenced by `std::any_of()`, and `std::is_partitioned()`.

3.54.2.28 `template<typename _ForwardIterator1, typename _ForwardIterator2 > _ForwardIterator1 std::search (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2 __last2) [inline]`

Search a sequence for a matching sub-sequence.

Parameters

<code>__first1</code>	A forward iterator.
<code>__last1</code>	A forward iterator.
<code>__first2</code>	A forward iterator.
<code>__last2</code>	A forward iterator.

Returns

The first iterator `i` in the range `[__first1,__last1-(__last2-__first2))` such that `*(i+N) == *(__first2+N)` for each `N` in the range `[0,__last2-__first2)`, or `__last1` if no such iterator exists.

Searches the range `[__first1,__last1)` for a sub-sequence that compares equal value-by-value with the sequence given by `[__first2,__last2)` and returns an iterator to the first element of the sub-sequence, or `__last1` if the sub-sequence is not found.

Because the sub-sequence must lie completely within the range `[__first1,__last1)` it must start at a position less than `__last1-(__last2-__first2)` where `__last2-__first2` is the length of the sub-sequence.

This means that the returned iterator `i` will be in the range `[__first1,__last1-(__last2-__first2))`

Definition at line 4024 of file `stl_algo.h`.

3.54.2.29 `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate > _ForwardIterator1 std::search (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2 __last2, _BinaryPredicate __predicate) [inline]`

Search a sequence for a matching sub-sequence using a predicate.

Parameters

<code>__first1</code>	A forward iterator.
<code>__last1</code>	A forward iterator.
<code>__first2</code>	A forward iterator.
<code>__last2</code>	A forward iterator.
<code>__predicate</code>	A binary predicate.

Returns

The first iterator `i` in the range `[__first1, __last1 - (__last2 - __first2))` such that `__predicate(*(i+N), *(__first2+N))` is true for each `N` in the range `[0, __last2 - __first2)`, or `__last1` if no such iterator exists.

Searches the range `[__first1, __last1)` for a sub-sequence that compares equal value-by-value with the sequence given by `[__first2, __last2)`, using `__predicate` to determine equality, and returns an iterator to the first element of the sub-sequence, or `__last1` if no such iterator exists.

See also

`search(_ForwardIter1, _ForwardIter1, _ForwardIter2, _ForwardIter2)`

Definition at line 4064 of file `stl_algo.h`.

3.54.2.30 `template<typename _ForwardIterator, typename _Integer, typename _Tp> _ForwardIterator std::search_n (`
`_ForwardIterator __first, _ForwardIterator __last, _Integer __count, const _Tp & __val) [inline]`

Search a sequence for a number of consecutive values.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__count</code>	The number of consecutive values.
<code>__val</code>	The value to find.

Returns

The first iterator `i` in the range `[__first, __last - __count)` such that `*(i+N) == __val` for each `N` in the range `[0, __count)`, or `__last` if no such iterator exists.

Searches the range `[__first, __last)` for `count` consecutive elements equal to `__val`.

Definition at line 4098 of file `stl_algo.h`.

3.54.2.31 `template<typename _ForwardIterator, typename _Integer, typename _Tp, typename _BinaryPredicate> _ForwardIterator`
`std::search_n (_ForwardIterator __first, _ForwardIterator __last, _Integer __count, const _Tp & __val, _BinaryPredicate`
`__binary_pred) [inline]`

Search a sequence for a number of consecutive values using a predicate.

Parameters

<code>__first</code>	A forward iterator.
<code>__last</code>	A forward iterator.
<code>__count</code>	The number of consecutive values.
<code>__val</code>	The value to find.
<code>__binary_pred</code>	A binary predicate.

Returns

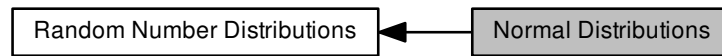
The first iterator `i` in the range `[__first, __last - __count)` such that `__binary_pred(*(i+N), __val)` is true for each `N` in the range `[0, __count)`, or `__last` if no such iterator exists.

Searches the range `[__first, __last)` for `__count` consecutive elements for which the predicate returns true.

Definition at line 4132 of file `stl_algo.h`.

3.55 Normal Distributions

Collaboration diagram for Normal Distributions:



Classes

- class `std::cauchy_distribution< _RealType >`
- class `std::chi_squared_distribution< _RealType >`
- class `std::fisher_f_distribution< _RealType >`
- class `std::gamma_distribution< _RealType >`
- class `std::lognormal_distribution< _RealType >`
- class `std::normal_distribution< _RealType >`
- class `std::student_t_distribution< _RealType >`

Functions

- template<typename _RealType >
bool `std::operator!=` (const `std::normal_distribution< _RealType >` &__d1, const `std::normal_distribution< _RealType >` &__d2)
- template<typename _RealType >
bool `std::operator!=` (const `std::lognormal_distribution< _RealType >` &__d1, const `std::lognormal_distribution< _RealType >` &__d2)
- template<typename _RealType >
bool `std::operator!=` (const `std::gamma_distribution< _RealType >` &__d1, const `std::gamma_distribution< _RealType >` &__d2)
- template<typename _RealType >
bool `std::operator!=` (const `std::chi_squared_distribution< _RealType >` &__d1, const `std::chi_squared_distribution< _RealType >` &__d2)
- template<typename _RealType >
bool `std::operator!=` (const `std::cauchy_distribution< _RealType >` &__d1, const `std::cauchy_distribution< _RealType >` &__d2)
- template<typename _RealType >
bool `std::operator!=` (const `std::fisher_f_distribution< _RealType >` &__d1, const `std::fisher_f_distribution< _RealType >` &__d2)
- template<typename _RealType >
bool `std::operator!=` (const `std::student_t_distribution< _RealType >` &__d1, const `std::student_t_distribution< _RealType >` &__d2)
- template<typename _RealType, typename _CharT, typename _Traits >
`std::basic_ostream< _CharT, _Traits >` & `std::operator<<` (`std::basic_ostream< _CharT, _Traits >` &__os, const `std::cauchy_distribution< _RealType >` &__x)
- template<typename _RealType, typename _CharT, typename _Traits >
`std::basic_istream< _CharT, _Traits >` & `std::operator>>` (`std::basic_istream< _CharT, _Traits >` &__is, `std::cauchy_distribution< _RealType >` &__x)

3.55.1 Detailed Description

3.55.2 Function Documentation

3.55.2.1 `template<typename _RealType> bool std::operator!=(const std::normal_distribution< _RealType> & __d1, const std::normal_distribution< _RealType> & __d2) [inline]`

Return true if two normal distributions are different.

Definition at line 2118 of file random.h.

3.55.2.2 `template<typename _RealType> bool std::operator!=(const std::lognormal_distribution< _RealType> & __d1, const std::lognormal_distribution< _RealType> & __d2) [inline]`

Return true if two lognormal distributions are different.

Definition at line 2322 of file random.h.

3.55.2.3 `template<typename _RealType> bool std::operator!=(const std::gamma_distribution< _RealType> & __d1, const std::gamma_distribution< _RealType> & __d2) [inline]`

Return true if two gamma distributions are different.

Definition at line 2542 of file random.h.

3.55.2.4 `template<typename _RealType> bool std::operator!=(const std::chi_squared_distribution< _RealType> & __d1, const std::chi_squared_distribution< _RealType> & __d2) [inline]`

Return true if two Chi-squared distributions are different.

Definition at line 2752 of file random.h.

3.55.2.5 `template<typename _RealType> bool std::operator!=(const std::cauchy_distribution< _RealType> & __d1, const std::cauchy_distribution< _RealType> & __d2) [inline]`

Return true if two Cauchy distributions have different parameters.

Definition at line 2919 of file random.h.

References `std::__detail::operator>>()`.

3.55.2.6 `template<typename _RealType> bool std::operator!=(const std::fisher_f_distribution< _RealType> & __d1, const std::fisher_f_distribution< _RealType> & __d2) [inline]`

Return true if two Fisher f distributions are different.

Definition at line 3175 of file random.h.

3.55.2.7 `template<typename _RealType> bool std::operator!=(const std::student_t_distribution< _RealType> & __d1, const std::student_t_distribution< _RealType> & __d2) [inline]`

Return true if two Student t distributions are different.

Definition at line 3388 of file random.h.

3.55.2.8 `template<typename _RealType, typename _CharT, typename _Traits> std::basic_ostream< _CharT, _Traits> & std::operator<< (std::basic_ostream< _CharT, _Traits> & __os, const std::cauchy_distribution< _RealType> & __x)`

Inserts a `cauchy_distribution` random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>cauchy_distribution</code> random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

Definition at line 2123 of file `bits/random.tcc`.

References `std::ios_base::flags()`, `std::left()`, and `std::scientific()`.

```
3.55.2.9 template<typename _RealType, typename _CharT, typename _Traits> std::basic_istream<_CharT, _Traits> &
std::operator>>( std::basic_istream<_CharT, _Traits> & __is, std::cauchy_distribution<_RealType> & __x
)
```

Extracts a `cauchy_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>cauchy_distribution</code> random number generator engine.

Returns

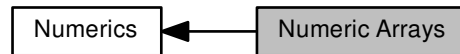
The input stream with `__x` extracted or in an error state.

Definition at line 2147 of file `bits/random.tcc`.

References `std::dec()`, `std::basic_ios<_CharT, _Traits>::fill()`, `std::ios_base::flags()`, `std::left()`, `std::gamma_distribution<_RealType>::operator()`, `std::__detail::operator>>()`, `std::cauchy_distribution<_RealType>::param()`, `std::fisher_f_distribution<_RealType>::param()`, `std::student_t_distribution<_RealType>::param()`, `std::ios_base::precision()`, `std::scientific()`, `std::skipws()`, `std::sqrt()`, and `std::basic_ios<_CharT, _Traits>::widen()`.

3.56 Numeric Arrays

Collaboration diagram for Numeric Arrays:



Classes

- class [std::gslice](#)
- class [std::gslice_array< _Tp >](#)
- class [std::indirect_array< _Tp >](#)
- class [std::mask_array< _Tp >](#)
- class [std::slice](#)
- class [std::slice_array< _Tp >](#)
- class [std::valarray< _Tp >](#)

Macros

- `#define _DEFINE_BINARY_OPERATOR(_Op, _Name)`
- `#define _DEFINE_VALARRAY_AUGMENTED_ASSIGNMENT(_Op, _Name)`
- `#define _DEFINE_VALARRAY_EXPR_AUGMENTED_ASSIGNMENT(_Op, _Name)`
- `#define _DEFINE_VALARRAY_OPERATOR(_Op, _Name)`
- `#define _DEFINE_VALARRAY_OPERATOR(_Op, _Name)`
- `#define _DEFINE_VALARRAY_OPERATOR(_Op, _Name)`
- `#define _DEFINE_VALARRAY_OPERATOR(_Op, _Name)`
- `#define _DEFINE_VALARRAY_OPERATOR(_Op, _Name)`
- `#define _DEFINE_VALARRAY_UNARY_OPERATOR(_Op, _Name)`

Functions

- [std::gslice::gslice](#) ()
- [std::gslice::gslice](#) (size_t __o, const valarray< size_t > & __l, const valarray< size_t > & __s)
- [std::gslice::gslice](#) (const gslice &)
- [std::gslice_array< _Tp >::gslice_array](#) (const gslice_array &)
- [std::indirect_array< _Tp >::indirect_array](#) (const indirect_array &)
- [std::mask_array< _Tp >::mask_array](#) (const mask_array &)
- [std::slice::slice](#) ()
- [std::slice::slice](#) (size_t __o, size_t __d, size_t __s)
- [std::slice_array< _Tp >::slice_array](#) (const slice_array &)
- [std::valarray< _Tp >::valarray](#) ()
- [std::valarray< _Tp >::valarray](#) (size_t)

- `std::valarray<_Tp>::valarray` (const _Tp &, size_t)
- `std::valarray<_Tp>::valarray` (const valarray &)
- `std::valarray<_Tp>::valarray` (valarray &&) noexcept
- `std::valarray<_Tp>::valarray` (const slice_array<_Tp> &)
- `std::valarray<_Tp>::valarray` (const gslice_array<_Tp> &)
- `std::valarray<_Tp>::valarray` (const mask_array<_Tp> &)
- `std::valarray<_Tp>::valarray` (const indirect_array<_Tp> &)
- `std::valarray<_Tp>::valarray` (initializer_list<_Tp>)
- template<class _Dom>
 - `std::valarray<_Tp>::valarray` (const _Expr<_Dom, _Tp> &__e)
- template<typename _Tp>
 - `std::valarray<_Tp>::valarray` (const _Tp *__restrict __p, size_t __n)
- `std::gslice::~gslice` ()
- `_Expr<_ValFunClos<_ValArray, _Tp>, _Tp> std::valarray<_Tp>::apply` (_Tp func(_Tp)) const
- `_Expr<_RefFunClos<_ValArray, _Tp>, _Tp> std::valarray<_Tp>::apply` (_Tp func(const _Tp &)) const
- template<class _Tp>
 - `_Tp * std::begin` (valarray<_Tp> &__va)
- template<class _Tp>
 - `const _Tp * std::begin` (const valarray<_Tp> &__va)
- `valarray<_Tp> std::valarray<_Tp>::cshift` (int __n) const
- template<class _Tp>
 - `_Tp * std::end` (valarray<_Tp> &__va)
- template<class _Tp>
 - `const _Tp * std::end` (const valarray<_Tp> &__va)
- `_Tp std::valarray<_Tp>::max` () const
- `_Tp std::valarray<_Tp>::min` () const
- `_UnaryOp<__logical_not>::Rt std::valarray<_Tp>::operator!` () const
- template<typename _Tp>
 - `_Expr<_BinClos<__not_equal_to, _ValArray, _Constant, _Tp, _Tp>, typename __fun<__not_equal_to, _Tp>::result_type> std::operator!=` (const valarray<_Tp> &__v, const _Tp &__t)
- template<typename _Tp>
 - `_Expr<_BinClos<__not_equal_to, _Constant, _ValArray, _Tp, _Tp>, typename __fun<__not_equal_to, _Tp>::result_type> std::operator!=` (const _Tp &__t, const valarray<_Tp> &__v)
- template<typename _Tp>
 - `_Expr<_BinClos<__not_equal_to, _ValArray, _ValArray, _Tp, _Tp>, typename __fun<__not_equal_to, _Tp>::result_type> std::operator!=` (const valarray<_Tp> &__v, const valarray<_Tp> &__w)
- template<typename _Tp>
 - `_Expr<_BinClos<__modulus, _ValArray, _ValArray, _Tp, _Tp>, typename __fun<__modulus, _Tp>::result_type> std::operator%` (const valarray<_Tp> &__v, const valarray<_Tp> &__w)
- template<typename _Tp>
 - `_Expr<_BinClos<__modulus, _ValArray, _Constant, _Tp, _Tp>, typename __fun<__modulus, _Tp>::result_type> std::operator%` (const valarray<_Tp> &__v, const _Tp &__t)
- template<typename _Tp>
 - `_Expr<_BinClos<__modulus, _Constant, _ValArray, _Tp, _Tp>, typename __fun<__modulus, _Tp>::result_type> std::operator%` (const _Tp &__t, const valarray<_Tp> &__v)
- void `std::gslice_array<_Tp>::operator%=` (const valarray<_Tp> &) const
- void `std::mask_array<_Tp>::operator%=` (const valarray<_Tp> &) const
- void `std::indirect_array<_Tp>::operator%=` (const valarray<_Tp> &) const
- template<class _Dom>
 - void `std::gslice_array<_Tp>::operator%=` (const _Expr<_Dom, _Tp> &) const
- template<class _Dom>
 - void `std::indirect_array<_Tp>::operator%=` (const _Expr<_Dom, _Tp> &) const

- `template<class _Dom >`
`void std::mask_array<_Tp>::operator%=(const _Expr<_Dom, _Tp> &) const`
- `void std::slice_array<_Tp>::operator%=(const valarray<_Tp> &) const`
- `template<class _Dom >`
`void std::slice_array<_Tp>::operator%=(const _Expr<_Dom, _Tp> &) const`
- `valarray<_Tp> & std::valarray<_Tp>::operator%=(const _Tp &)`
- `valarray<_Tp> & std::valarray<_Tp>::operator%=(const valarray<_Tp> &)`
- `template<class _Dom >`
`valarray<_Tp> & std::valarray<_Tp>::operator%=(const _Expr<_Dom, _Tp> &)`
- `template<typename _Tp >`
`_Expr<_BinClos<__bitwise_and, _ValArray, _Constant, _Tp, _Tp>, typename __fun<__bitwise_and, _Tp>::result_type> std::operator&(const valarray<_Tp> &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr<_BinClos<__bitwise_and, _Constant, _ValArray, _Tp, _Tp>, typename __fun<__bitwise_and, _Tp>::result_type> std::operator&(const _Tp &__t, const valarray<_Tp> &__v)`
- `template<typename _Tp >`
`_Expr<_BinClos<__bitwise_and, _ValArray, _ValArray, _Tp, _Tp>, typename __fun<__bitwise_and, _Tp>::result_type> std::operator&(const valarray<_Tp> &__v, const valarray<_Tp> &__w)`
- `template<typename _Tp >`
`_Expr<_BinClos<__logical_and, _Constant, _ValArray, _Tp, _Tp>, typename __fun<__logical_and, _Tp>::result_type> std::operator&&(const _Tp &__t, const valarray<_Tp> &__v)`
- `template<typename _Tp >`
`_Expr<_BinClos<__logical_and, _ValArray, _Constant, _Tp, _Tp>, typename __fun<__logical_and, _Tp>::result_type> std::operator&&(const valarray<_Tp> &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr<_BinClos<__logical_and, _ValArray, _ValArray, _Tp, _Tp>, typename __fun<__logical_and, _Tp>::result_type> std::operator&&(const valarray<_Tp> &__v, const valarray<_Tp> &__w)`
- `void std::gslice_array<_Tp>::operator&=(const valarray<_Tp> &) const`
- `void std::mask_array<_Tp>::operator&=(const valarray<_Tp> &) const`
- `void std::indirect_array<_Tp>::operator&=(const valarray<_Tp> &) const`
- `template<class _Dom >`
`void std::gslice_array<_Tp>::operator&=(const _Expr<_Dom, _Tp> &) const`
- `template<class _Dom >`
`void std::indirect_array<_Tp>::operator&=(const _Expr<_Dom, _Tp> &) const`
- `template<class _Dom >`
`void std::mask_array<_Tp>::operator&=(const _Expr<_Dom, _Tp> &) const`
- `void std::slice_array<_Tp>::operator&=(const valarray<_Tp> &) const`
- `template<class _Dom >`
`void std::slice_array<_Tp>::operator&=(const _Expr<_Dom, _Tp> &) const`
- `valarray<_Tp> & std::valarray<_Tp>::operator&=(const _Tp &)`
- `valarray<_Tp> & std::valarray<_Tp>::operator&=(const valarray<_Tp> &)`
- `template<class _Dom >`
`valarray<_Tp> & std::valarray<_Tp>::operator&=(const _Expr<_Dom, _Tp> &)`
- `template<typename _Tp >`
`_Expr<_BinClos<__multiplies, _ValArray, _Constant, _Tp, _Tp>, typename __fun<__multiplies, _Tp>::result_type> std::operator*(const valarray<_Tp> &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr<_BinClos<__multiplies, _Constant, _ValArray, _Tp, _Tp>, typename __fun<__multiplies, _Tp>::result_type> std::operator*(const _Tp &__t, const valarray<_Tp> &__v)`
- `template<typename _Tp >`
`_Expr<_BinClos<__multiplies, _ValArray, _ValArray, _Tp, _Tp>, typename __fun<__multiplies, _Tp>::result_type> std::operator*(const valarray<_Tp> &__v, const valarray<_Tp> &__w)`

- void `std::gslice_array<_Tp>::operator*=` (const valarray<_Tp> &) const
- void `std::mask_array<_Tp>::operator*=` (const valarray<_Tp> &) const
- void `std::indirect_array<_Tp>::operator*=` (const valarray<_Tp> &) const
- template<class _Dom>
void `std::gslice_array<_Tp>::operator*=` (const _Expr<_Dom, _Tp> &) const
- template<class _Dom>
void `std::indirect_array<_Tp>::operator*=` (const _Expr<_Dom, _Tp> &) const
- template<class _Dom>
void `std::mask_array<_Tp>::operator*=` (const _Expr<_Dom, _Tp> &) const
- void `std::slice_array<_Tp>::operator*=` (const valarray<_Tp> &) const
- template<class _Dom>
void `std::slice_array<_Tp>::operator*=` (const _Expr<_Dom, _Tp> &) const
- valarray<_Tp> & `std::valarray<_Tp>::operator*=` (const _Tp &)
- valarray<_Tp> & `std::valarray<_Tp>::operator*=` (const valarray<_Tp> &)
- template<class _Dom>
valarray<_Tp> & `std::valarray<_Tp>::operator*=` (const _Expr<_Dom, _Tp> &)
- _UnaryOp<__unary_plus>::Rt `std::valarray<_Tp>::operator+` () const
- template<typename _Tp>
_Expr<__BinClos<__plus, _ValArray, _Constant, _Tp, _Tp>, typename __fun<__plus, _Tp>::result_type>
`std::operator+` (const valarray<_Tp> &__v, const _Tp &__t)
- template<typename _Tp>
_Expr<__BinClos<__plus, _ValArray, _ValArray, _Tp, _Tp>, typename __fun<__plus, _Tp>::result_type>
`std::operator+` (const valarray<_Tp> &__v, const valarray<_Tp> &__w)
- template<typename _Tp>
_Expr<__BinClos<__plus, _Constant, _ValArray, _Tp, _Tp>, typename __fun<__plus, _Tp>::result_type>
`std::operator+` (const _Tp &__t, const valarray<_Tp> &__v)
- void `std::gslice_array<_Tp>::operator+=` (const valarray<_Tp> &) const
- void `std::mask_array<_Tp>::operator+=` (const valarray<_Tp> &) const
- void `std::indirect_array<_Tp>::operator+=` (const valarray<_Tp> &) const
- template<class _Dom>
void `std::gslice_array<_Tp>::operator+=` (const _Expr<_Dom, _Tp> &) const
- template<class _Dom>
void `std::indirect_array<_Tp>::operator+=` (const _Expr<_Dom, _Tp> &) const
- template<class _Dom>
void `std::mask_array<_Tp>::operator+=` (const _Expr<_Dom, _Tp> &) const
- void `std::slice_array<_Tp>::operator+=` (const valarray<_Tp> &) const
- template<class _Dom>
void `std::slice_array<_Tp>::operator+=` (const _Expr<_Dom, _Tp> &) const
- valarray<_Tp> & `std::valarray<_Tp>::operator+=` (const _Tp &)
- valarray<_Tp> & `std::valarray<_Tp>::operator+=` (const valarray<_Tp> &)
- template<class _Dom>
valarray<_Tp> & `std::valarray<_Tp>::operator+=` (const _Expr<_Dom, _Tp> &)
- _UnaryOp<__negate>::Rt `std::valarray<_Tp>::operator-` () const
- template<typename _Tp>
_Expr<__BinClos<__minus, _ValArray, _ValArray, _Tp, _Tp>, typename __fun<__minus, _Tp>::result_type>
> `std::operator-` (const valarray<_Tp> &__v, const valarray<_Tp> &__w)
- template<typename _Tp>
_Expr<__BinClos<__minus, _ValArray, _Constant, _Tp, _Tp>, typename __fun<__minus, _Tp>::result_type>
> `std::operator-` (const valarray<_Tp> &__v, const _Tp &__t)
- template<typename _Tp>
_Expr<__BinClos<__minus, _Constant, _ValArray, _Tp, _Tp>, typename __fun<__minus, _Tp>::result_type>
> `std::operator-` (const _Tp &__t, const valarray<_Tp> &__v)

- void `std::gslice_array<_Tp>::operator=` (const valarray<_Tp> &) const
- void `std::mask_array<_Tp>::operator=` (const valarray<_Tp> &) const
- void `std::indirect_array<_Tp>::operator=` (const valarray<_Tp> &) const
- template<class _Dom>
void `std::gslice_array<_Tp>::operator=` (const _Expr<_Dom, _Tp> &) const
- template<class _Dom>
void `std::indirect_array<_Tp>::operator=` (const _Expr<_Dom, _Tp> &) const
- template<class _Dom>
void `std::mask_array<_Tp>::operator=` (const _Expr<_Dom, _Tp> &) const
- void `std::slice_array<_Tp>::operator=` (const valarray<_Tp> &) const
- template<class _Dom>
void `std::slice_array<_Tp>::operator=` (const _Expr<_Dom, _Tp> &) const
- valarray<_Tp> & `std::valarray<_Tp>::operator=` (const _Tp &)
- valarray<_Tp> & `std::valarray<_Tp>::operator=` (const valarray<_Tp> &)
- template<class _Dom>
valarray<_Tp> & `std::valarray<_Tp>::operator=` (const _Expr<_Dom, _Tp> &)
- template<typename _Tp>
_Expr<_BinClos<__divides, _ValArray, _ValArray, _Tp, _Tp>, typename __fun<__divides, _Tp>::result_type>
> `std::operator/` (const valarray<_Tp> &__v, const valarray<_Tp> &__w)
- template<typename _Tp>
_Expr<_BinClos<__divides, _ValArray, _Constant, _Tp, _Tp>, typename __fun<__divides, _Tp>::result_type>
type> `std::operator/` (const valarray<_Tp> &__v, const _Tp &__t)
- template<typename _Tp>
_Expr<_BinClos<__divides, _Constant, _ValArray, _Tp, _Tp>, typename __fun<__divides, _Tp>::result_type>
type> `std::operator/` (const _Tp &__t, const valarray<_Tp> &__v)
- void `std::gslice_array<_Tp>::operator/=` (const valarray<_Tp> &) const
- void `std::mask_array<_Tp>::operator/=` (const valarray<_Tp> &) const
- void `std::indirect_array<_Tp>::operator/=` (const valarray<_Tp> &) const
- template<class _Dom>
void `std::gslice_array<_Tp>::operator/=` (const _Expr<_Dom, _Tp> &) const
- template<class _Dom>
void `std::mask_array<_Tp>::operator/=` (const _Expr<_Dom, _Tp> &) const
- template<class _Dom>
void `std::indirect_array<_Tp>::operator/=` (const _Expr<_Dom, _Tp> &) const
- void `std::slice_array<_Tp>::operator/=` (const valarray<_Tp> &) const
- template<class _Dom>
void `std::slice_array<_Tp>::operator/=` (const _Expr<_Dom, _Tp> &) const
- valarray<_Tp> & `std::valarray<_Tp>::operator/=` (const _Tp &)
- valarray<_Tp> & `std::valarray<_Tp>::operator/=` (const valarray<_Tp> &)
- template<class _Dom>
valarray<_Tp> & `std::valarray<_Tp>::operator/=` (const _Expr<_Dom, _Tp> &)
- template<typename _Tp>
_Expr<_BinClos<__less, _ValArray, _ValArray, _Tp, _Tp>, typename __fun<__less, _Tp>::result_type>
> `std::operator<` (const valarray<_Tp> &__v, const valarray<_Tp> &__w)
- template<typename _Tp>
_Expr<_BinClos<__less, _ValArray, _Constant, _Tp, _Tp>, typename __fun<__less, _Tp>::result_type>
> `std::operator<` (const valarray<_Tp> &__v, const _Tp &__t)
- template<typename _Tp>
_Expr<_BinClos<__less, _Constant, _ValArray, _Tp, _Tp>, typename __fun<__less, _Tp>::result_type>
> `std::operator<` (const _Tp &__t, const valarray<_Tp> &__v)
- template<typename _Tp>
_Expr<_BinClos<__shift_left, _Constant, _ValArray, _Tp, _Tp>, typename __fun<__shift_left, _Tp>::result_type>
type> `std::operator<<` (const _Tp &__t, const valarray<_Tp> &__v)

- `template<typename _Tp >`
`_Expr< __BinClos< __shift_left, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __shift_left, _Tp >::result_t>`
`_type > std::operator<< (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< __BinClos< __shift_left, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __shift_left, _Tp >::result_t>`
`_type > std::operator<< (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `void std::gslice_array< _Tp >::operator<<= (const valarray< _Tp > &) const`
- `void std::mask_array< _Tp >::operator<<= (const valarray< _Tp > &) const`
- `void std::indirect_array< _Tp >::operator<<= (const valarray< _Tp > &) const`
- `template<class _Dom >`
`void std::gslice_array< _Tp >::operator<<= (const _Expr< _Dom, _Tp > &) const`
- `template<class _Dom >`
`void std::indirect_array< _Tp >::operator<<= (const _Expr< _Dom, _Tp > &) const`
- `template<class _Dom >`
`void std::mask_array< _Tp >::operator<<= (const _Expr< _Dom, _Tp > &) const`
- `void std::slice_array< _Tp >::operator<<= (const valarray< _Tp > &) const`
- `template<class _Dom >`
`void std::slice_array< _Tp >::operator<<= (const _Expr< _Dom, _Tp > &) const`
- `valarray< _Tp > & std::valarray< _Tp >::operator<<= (const _Tp &)`
- `valarray< _Tp > & std::valarray< _Tp >::operator<<= (const valarray< _Tp > &)`
- `template<class _Dom >`
`valarray< _Tp > & std::valarray< _Tp >::operator<<= (const _Expr< _Dom, _Tp > &)`
- `template<typename _Tp >`
`_Expr< __BinClos< __less_equal, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __less_equal, _Tp >::result_type >`
`std::operator<= (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< __BinClos< __less_equal, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __less_equal, _Tp >::result_type >`
`std::operator<= (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< __BinClos< __less_equal, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __less_equal, _Tp >::result_type >`
`std::operator<= (const valarray< _Tp > &__v, const _Tp &__t)`
- `gslice_array & std::gslice_array< _Tp >::operator= (const gslice_array &)`
- `indirect_array & std::indirect_array< _Tp >::operator= (const indirect_array &)`
- `mask_array & std::mask_array< _Tp >::operator= (const mask_array &)`
- `void std::gslice_array< _Tp >::operator= (const valarray< _Tp > &) const`
- `void std::mask_array< _Tp >::operator= (const valarray< _Tp > &) const`
- `void std::indirect_array< _Tp >::operator= (const valarray< _Tp > &) const`
- `gslice & std::gslice::operator= (const gslice &)`
- `void std::gslice_array< _Tp >::operator= (const _Tp &) const`
- `void std::mask_array< _Tp >::operator= (const _Tp &) const`
- `void std::indirect_array< _Tp >::operator= (const _Tp &) const`
- `template<class _Dom >`
`void std::gslice_array< _Tp >::operator= (const _Expr< _Dom, _Tp > &) const`
- `template<class _Dom >`
`void std::indirect_array< _Tp >::operator= (const _Expr< _Dom, _Tp > &) const`
- `slice_array & std::slice_array< _Tp >::operator= (const slice_array &)`
- `void std::slice_array< _Tp >::operator= (const valarray< _Tp > &) const`
- `void std::slice_array< _Tp >::operator= (const _Tp &) const`
- `template<class _Dom >`
`void std::slice_array< _Tp >::operator= (const _Expr< _Dom, _Tp > &) const`
- `template<class _Ex >`
`void std::mask_array< _Tp >::operator= (const _Expr< _Ex, _Tp > &__e) const`

- `valarray<_Tp> & std::valarray<_Tp>::operator=` (const valarray<_Tp> &__v)
- `valarray<_Tp> & std::valarray<_Tp>::operator=` (valarray<_Tp> &&__v) noexcept
- `valarray<_Tp> & std::valarray<_Tp>::operator=` (const _Tp &__t)
- `valarray<_Tp> & std::valarray<_Tp>::operator=` (const slice_array<_Tp> &__sa)
- `valarray<_Tp> & std::valarray<_Tp>::operator=` (const gslice_array<_Tp> &__ga)
- `valarray<_Tp> & std::valarray<_Tp>::operator=` (const mask_array<_Tp> &__ma)
- `valarray<_Tp> & std::valarray<_Tp>::operator=` (const indirect_array<_Tp> &__ia)
- `valarray & std::valarray<_Tp>::operator=` (initializer_list<_Tp> __l)
- `template<class _Dom>`
`valarray<_Tp> & std::valarray<_Tp>::operator=` (const _Expr<_Dom, _Tp> &)
- `template<typename _Tp>`
`_Expr<_BinClos<__equal_to, _ValArray, _ValArray, _Tp, _Tp>, typename __fun<__equal_to, _Tp>::result_type>`
`std::operator==` (const valarray<_Tp> &__v, const valarray<_Tp> &__w)
- `template<typename _Tp>`
`_Expr<_BinClos<__equal_to, _Constant, _ValArray, _Tp, _Tp>, typename __fun<__equal_to, _Tp>::result_type>`
`std::operator==` (const _Tp &__t, const valarray<_Tp> &__v)
- `template<typename _Tp>`
`_Expr<_BinClos<__equal_to, _ValArray, _Constant, _Tp, _Tp>, typename __fun<__equal_to, _Tp>::result_type>`
`std::operator==` (const valarray<_Tp> &__v, const _Tp &__t)
- `template<typename _Tp>`
`_Expr<_BinClos<__greater, _ValArray, _ValArray, _Tp, _Tp>, typename __fun<__greater, _Tp>::result_type>`
`std::operator>` (const valarray<_Tp> &__v, const valarray<_Tp> &__w)
- `template<typename _Tp>`
`_Expr<_BinClos<__greater, _ValArray, _Constant, _Tp, _Tp>, typename __fun<__greater, _Tp>::result_type>`
`std::operator>` (const valarray<_Tp> &__v, const _Tp &__t)
- `template<typename _Tp>`
`_Expr<_BinClos<__greater, _Constant, _ValArray, _Tp, _Tp>, typename __fun<__greater, _Tp>::result_type>`
`std::operator>` (const _Tp &__t, const valarray<_Tp> &__v)
- `template<typename _Tp>`
`_Expr<_BinClos<__greater_equal, _Constant, _ValArray, _Tp, _Tp>, typename __fun<__greater_equal, _Tp>::result_type>`
`std::operator>=` (const _Tp &__t, const valarray<_Tp> &__v)
- `template<typename _Tp>`
`_Expr<_BinClos<__greater_equal, _ValArray, _Constant, _Tp, _Tp>, typename __fun<__greater_equal, _Tp>::result_type>`
`std::operator>=` (const valarray<_Tp> &__v, const _Tp &__t)
- `template<typename _Tp>`
`_Expr<_BinClos<__greater_equal, _ValArray, _ValArray, _Tp, _Tp>, typename __fun<__greater_equal, _Tp>::result_type>`
`std::operator>=` (const valarray<_Tp> &__v, const valarray<_Tp> &__w)
- `template<typename _Tp>`
`_Expr<_BinClos<__shift_right, _ValArray, _Constant, _Tp, _Tp>, typename __fun<__shift_right, _Tp>::result_type>`
`std::operator>>` (const valarray<_Tp> &__v, const _Tp &__t)
- `template<typename _Tp>`
`_Expr<_BinClos<__shift_right, _Constant, _ValArray, _Tp, _Tp>, typename __fun<__shift_right, _Tp>::result_type>`
`std::operator>>` (const _Tp &__t, const valarray<_Tp> &__v)
- `template<typename _Tp>`
`_Expr<_BinClos<__shift_right, _ValArray, _ValArray, _Tp, _Tp>, typename __fun<__shift_right, _Tp>::result_type>`
`std::operator>>` (const valarray<_Tp> &__v, const valarray<_Tp> &__w)
- `void std::gslice_array<_Tp>::operator>>=` (const valarray<_Tp> &) const
- `void std::mask_array<_Tp>::operator>>=` (const valarray<_Tp> &) const
- `void std::indirect_array<_Tp>::operator>>=` (const valarray<_Tp> &) const
- `template<class _Dom>`
`void std::gslice_array<_Tp>::operator>>=` (const _Expr<_Dom, _Tp> &) const
- `template<class _Dom>`
`void std::indirect_array<_Tp>::operator>>=` (const _Expr<_Dom, _Tp> &) const

- `template<class _Dom >`
`void std::mask_array<_Tp>::operator>>= (const _Expr<_Dom, _Tp> &) const`
- `void std::slice_array<_Tp>::operator>>= (const valarray<_Tp> &) const`
- `template<class _Dom >`
`void std::slice_array<_Tp>::operator>>= (const _Expr<_Dom, _Tp> &) const`
- `valarray<_Tp> & std::valarray<_Tp>::operator>>= (const _Tp &)`
- `valarray<_Tp> & std::valarray<_Tp>::operator>>= (const valarray<_Tp> &)`
- `template<class _Dom >`
`valarray<_Tp> & std::valarray<_Tp>::operator>>= (const _Expr<_Dom, _Tp> &)`
- `_Tp & std::valarray<_Tp>::operator[] (size_t __i)`
- `const _Tp & std::valarray<_Tp>::operator[] (size_t) const`
- `_Expr<_SClos<_ValArray, _Tp>, _Tp> std::valarray<_Tp>::operator[] (slice __s) const`
- `slice_array<_Tp> std::valarray<_Tp>::operator[] (slice __s)`
- `_Expr<_GClos<_ValArray, _Tp>, _Tp> std::valarray<_Tp>::operator[] (const gslice &__s) const`
- `gslice_array<_Tp> std::valarray<_Tp>::operator[] (const gslice &__s)`
- `valarray<_Tp> std::valarray<_Tp>::operator[] (const valarray<bool> &__m) const`
- `mask_array<_Tp> std::valarray<_Tp>::operator[] (const valarray<bool> &__m)`
- `_Expr<_IClos<_ValArray, _Tp>, _Tp> std::valarray<_Tp>::operator[] (const valarray<size_t> &__i) const`
- `indirect_array<_Tp> std::valarray<_Tp>::operator[] (const valarray<size_t> &__i)`
- `template<typename _Tp >`
`_Expr<_BinClos<__bitwise_xor, _ValArray, _ValArray, _Tp, _Tp>, typename __fun<__bitwise_xor, _Tp> <-
::result_type> std::operator^ (const valarray<_Tp> &__v, const valarray<_Tp> &__w)`
- `template<typename _Tp >`
`_Expr<_BinClos<__bitwise_xor, _ValArray, _Constant, _Tp, _Tp>, typename __fun<__bitwise_xor, _Tp> <-
::result_type> std::operator^ (const valarray<_Tp> &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr<_BinClos<__bitwise_xor, _Constant, _ValArray, _Tp, _Tp>, typename __fun<__bitwise_xor, _Tp> <-
::result_type> std::operator^ (const _Tp &__t, const valarray<_Tp> &__v)`
- `void std::gslice_array<_Tp>::operator^= (const valarray<_Tp> &) const`
- `void std::mask_array<_Tp>::operator^= (const valarray<_Tp> &) const`
- `void std::indirect_array<_Tp>::operator^= (const valarray<_Tp> &) const`
- `template<class _Dom >`
`void std::gslice_array<_Tp>::operator^= (const _Expr<_Dom, _Tp> &) const`
- `template<class _Dom >`
`void std::mask_array<_Tp>::operator^= (const _Expr<_Dom, _Tp> &) const`
- `template<class _Dom >`
`void std::indirect_array<_Tp>::operator^= (const _Expr<_Dom, _Tp> &) const`
- `void std::slice_array<_Tp>::operator^= (const valarray<_Tp> &) const`
- `template<class _Dom >`
`void std::slice_array<_Tp>::operator^= (const _Expr<_Dom, _Tp> &) const`
- `valarray<_Tp> & std::valarray<_Tp>::operator^= (const _Tp &)`
- `valarray<_Tp> & std::valarray<_Tp>::operator^= (const valarray<_Tp> &)`
- `template<class _Dom >`
`valarray<_Tp> & std::valarray<_Tp>::operator^= (const _Expr<_Dom, _Tp> &)`
- `template<typename _Tp >`
`_Expr<_BinClos<__bitwise_or, _Constant, _ValArray, _Tp, _Tp>, typename __fun<__bitwise_or, _Tp> <-
::result_type> std::operator| (const _Tp &__t, const valarray<_Tp> &__v)`
- `template<typename _Tp >`
`_Expr<_BinClos<__bitwise_or, _ValArray, _Constant, _Tp, _Tp>, typename __fun<__bitwise_or, _Tp> <-
::result_type> std::operator| (const valarray<_Tp> &__v, const _Tp &__t)`

- `template<typename _Tp >`
`_Expr< _BinClos< __bitwise_or, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __bitwise_or, _Tp >↔`
`::result_type > std::operator| (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `void std::gslice_array< _Tp >::operator|= (const valarray< _Tp > &) const`
- `void std::mask_array< _Tp >::operator|= (const valarray< _Tp > &) const`
- `void std::indirect_array< _Tp >::operator|= (const valarray< _Tp > &) const`
- `template<class _Dom >`
`void std::gslice_array< _Tp >::operator|= (const _Expr< _Dom, _Tp > &) const`
- `template<class _Dom >`
`void std::indirect_array< _Tp >::operator|= (const _Expr< _Dom, _Tp > &) const`
- `template<class _Dom >`
`void std::mask_array< _Tp >::operator|= (const _Expr< _Dom, _Tp > &) const`
- `void std::slice_array< _Tp >::operator|= (const valarray< _Tp > &) const`
- `template<class _Dom >`
`void std::slice_array< _Tp >::operator|= (const _Expr< _Dom, _Tp > &) const`
- `valarray< _Tp > & std::valarray< _Tp >::operator|= (const _Tp &)`
- `valarray< _Tp > & std::valarray< _Tp >::operator|= (const valarray< _Tp > &)`
- `template<class _Dom >`
`valarray< _Tp > & std::valarray< _Tp >::operator|= (const _Expr< _Dom, _Tp > &)`
- `template<typename _Tp >`
`_Expr< _BinClos< __logical_or, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __logical_or, _Tp >↔`
`::result_type > std::operator|| (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __logical_or, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __logical_or, _Tp >↔`
`::result_type > std::operator|| (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __logical_or, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __logical_or, _Tp >↔`
`::result_type > std::operator|| (const _Tp &__t, const valarray< _Tp > &__v)`
- `_UnaryOp< __bitwise_not >::Rt std::valarray< _Tp >::operator~ () const`
- `void std::valarray< _Tp >::resize (size_t __size, _Tp __c=_Tp())`
- `valarray< _Tp > std::valarray< _Tp >::shift (int __n) const`
- `size_t std::slice::size () const`
- `valarray< size_t > std::gslice::size () const`
- `size_t std::valarray< _Tp >::size () const`
- `size_t std::slice::start () const`
- `size_t std::gslice::start () const`
- `size_t std::slice::stride () const`
- `valarray< size_t > std::gslice::stride () const`
- `_Tp std::valarray< _Tp >::sum () const`
- `void std::valarray< _Tp >::swap (valarray< _Tp > &__v) noexcept`

3.56.1 Detailed Description

Classes and functions for representing and manipulating arrays of elements.

3.56.2 Function Documentation

3.56.2.1 `std::gslice::gslice ()` `[inline]`

Construct an empty slice.

Definition at line 149 of file `gslice.h`.

3.56.2.2 `std::gslice::gslice (size_t __o, const valarray< size_t > & __l, const valarray< size_t > & __s)` `[inline]`

Construct a slice.

Constructs a slice with as many dimensions as the length of the *l* and *s* arrays.

Parameters

<code>__o</code>	Offset in array of first element.
<code>__l</code>	Array of dimension lengths.
<code>__s</code>	Array of dimension strides between array elements.

Definition at line 153 of file `gslice.h`.

3.56.2.3 `std::gslice::gslice (const gslice & __g)` `[inline]`

Copy constructor.

Definition at line 158 of file `gslice.h`.

3.56.2.4 `template<typename _Tp> std::gslice_array< _Tp >::gslice_array (const gslice_array< _Tp > & __a)`
`[inline]`

Copy constructor. Both slices refer to the same underlying array.

Definition at line 143 of file `gslice_array.h`.

3.56.2.5 `template<typename _Tp> std::indirect_array< _Tp >::indirect_array (const indirect_array< _Tp > & __a)`
`[inline]`

Copy constructor. Both slices refer to the same underlying array.

Definition at line 143 of file `indirect_array.h`.

3.56.2.6 `template<typename _Tp> std::mask_array< _Tp >::mask_array (const mask_array< _Tp > & a)` `[inline]`

Copy constructor. Both slices refer to the same underlying array.

Definition at line 139 of file `mask_array.h`.

3.56.2.7 `std::slice::slice ()` `[inline]`

Construct an empty slice.

Definition at line 90 of file `slice_array.h`.

3.56.2.8 `std::slice::slice (size_t __o, size_t __d, size_t __s)` `[inline]`

Construct a slice.

Parameters

\leftarrow _o	Offset in array of first element.
\leftarrow _d	Number of elements in slice.
\leftarrow _s	Stride between array elements.

Definition at line 94 of file slice_array.h.

3.56.2.9 `template<typename _Tp> std::slice_array<_Tp>::slice_array (const slice_array<_Tp> &a) [inline]`

Copy constructor. Both slices refer to the same underlying array.

Definition at line 207 of file slice_array.h.

3.56.2.10 `template<typename _Tp> std::valarray<_Tp>::valarray () [inline]`

Construct an empty array.

Definition at line 605 of file valarray.

3.56.2.11 `template<typename _Tp> std::valarray<_Tp>::valarray (size_t n) [inline],[explicit]`

Construct an array with n elements.

Definition at line 609 of file valarray.

3.56.2.12 `template<typename _Tp> std::valarray<_Tp>::valarray (const _Tp &t, size_t n) [inline]`

Construct an array with n elements initialized to t .

Definition at line 615 of file valarray.

3.56.2.13 `template<typename _Tp> std::valarray<_Tp>::valarray (const valarray<_Tp> &v) [inline]`

Copy constructor.

Definition at line 630 of file valarray.

3.56.2.14 `template<typename _Tp> std::valarray<_Tp>::valarray (valarray<_Tp> &&v) [inline],
[noexcept]`

Move constructor.

Definition at line 638 of file valarray.

3.56.2.15 `template<typename _Tp> std::valarray<_Tp>::valarray (const slice_array<_Tp> &__sa) [inline]`

Construct an array with the same size and values in *sa*.

Definition at line 648 of file `valarray`.

3.56.2.16 `template<typename _Tp> std::valarray<_Tp>::valarray (const gslice_array<_Tp> &__ga) [inline]`

Construct an array with the same size and values in *ga*.

Definition at line 657 of file `valarray`.

3.56.2.17 `template<typename _Tp> std::valarray<_Tp>::valarray (const mask_array<_Tp> &__ma) [inline]`

Construct an array with the same size and values in *ma*.

Definition at line 668 of file `valarray`.

3.56.2.18 `template<typename _Tp> std::valarray<_Tp>::valarray (const indirect_array<_Tp> &__ia) [inline]`

Construct an array with the same size and values in *ia*.

Definition at line 677 of file `valarray`.

3.56.2.19 `template<typename _Tp> std::valarray<_Tp>::valarray (initializer_list<_Tp> __l) [inline]`

Construct an array with an `initializer_list` of values.

Definition at line 687 of file `valarray`.

3.56.2.20 `std::gslice::~gslice () [inline]`

Destructor.

Definition at line 163 of file `gslice.h`.

3.56.2.21 `template<class _Tp> _Expr<_ValFunClos<_ValArray, _Tp>, _Tp> std::valarray<_Tp>::apply (_Tp func_Tp) const [inline]`

Apply a function to the array.

Returns a new `valarray` with elements assigned to the result of applying `func` to the corresponding element of this array. The new array has the same size as this one.

Parameters

<i>func</i>	Function of <code>Tp</code> returning <code>Tp</code> to apply.
-------------	---

Returns

New valarray with transformed elements.

Definition at line 1049 of file valarray.

3.56.2.22 `template<class _Tp> _Expr<_RefFunClos<_ValArray, _Tp>, _Tp> std::valarray<_Tp>::apply (_Tp funcconst
_Tp &) const [inline]`

Apply a function to the array.

Returns a new valarray with elements assigned to the result of applying func to the corresponding element of this array. The new array has the same size as this one.

Parameters

<i>func</i>	Function of const Tp& returning Tp to apply.
-------------	--

Returns

New valarray with transformed elements.

Definition at line 1057 of file valarray.

3.56.2.23 `template<class _Tp> _Tp * std::begin (valarray<_Tp> &__va) [inline]`

Return an iterator pointing to the first element of the valarray.

Parameters

<code>__va</code>	valarray.
-------------------	-----------

Definition at line 1196 of file valarray.

3.56.2.24 `template<class _Tp> const _Tp * std::begin (const valarray<_Tp> &__va) [inline]`

Return an iterator pointing to the first element of the const valarray.

Parameters

<code>__va</code>	valarray.
-------------------	-----------

Definition at line 1206 of file valarray.

3.56.2.25 `template<class _Tp> valarray<_Tp> std::valarray<_Tp>::cshift (int __n) const [inline]`

Return a rotated array.

A new valarray is constructed as a copy of this array with elements in shifted positions. For an element with index i , the new position is $(i - n) \% \text{size}()$. The new valarray has the same size as the current one. Elements that are shifted beyond the array bounds are shifted into the other end of the array. No elements are lost.

Positive arguments shift toward index 0, wrapping around the top. Negative arguments shift towards the top, wrapping around to 0.

Parameters

$_n$	Number of element positions to rotate.
-------	--

Returns

New valarray with elements in shifted positions.

Definition at line 975 of file valarray.

3.56.2.26 `template<class _Tp> _Tp * std::end (valarray<_Tp> &__va) [inline]`

Return an iterator pointing to one past the last element of the valarray.

Parameters

<code>__va</code>	valarray.
-------------------	-----------

Definition at line 1216 of file valarray.

3.56.2.27 `template<class _Tp> const _Tp * std::end (const valarray<_Tp> &__va) [inline]`

Return an iterator pointing to one past the last element of the const valarray.

Parameters

<code>__va</code>	valarray.
-------------------	-----------

Definition at line 1226 of file valarray.

3.56.2.28 `template<typename _Tp> _Tp std::valarray<_Tp>::max () const [inline]`

Return the maximum element using operator<().

Definition at line 1041 of file valarray.

3.56.2.29 `template<typename _Tp> _Tp std::valarray<_Tp>::min () const [inline]`

Return the minimum element using operator<().

Definition at line 1033 of file valarray.

3.56.2.30 `template<typename _Tp> valarray<_Tp>::template _UnaryOp<__logical_not>::_Rt std::valarray<_Tp>::operator! () const [inline]`

Return a new valarray by applying unary ! to each element.

Definition at line 1076 of file valarray.

3.56.2.31 `template<typename _Tp> void std::gslice_array<_Tp>::operator%=(const valarray<_Tp> &__v) const [inline]`

Modulo slice elements by corresponding elements of *v*.

Definition at line 202 of file gslice_array.h.

3.56.2.32 `template<typename _Tp> void std::mask_array<_Tp>::operator%=(const valarray<_Tp> &__v) const [inline]`

Modulo slice elements by corresponding elements of *v*.

Definition at line 192 of file mask_array.h.

3.56.2.33 `template<typename _Tp> void std::indirect_array<_Tp>::operator%=(const valarray<_Tp> &__v) const [inline]`

Modulo slice elements by corresponding elements of *v*.

Definition at line 196 of file indirect_array.h.

3.56.2.34 `template<typename _Tp> void std::slice_array<_Tp>::operator%=(const valarray<_Tp> &__v) const [inline]`

Modulo slice elements by corresponding elements of *v*.

Definition at line 258 of file slice_array.h.

3.56.2.35 `template<class _Tp> valarray<_Tp> & std::valarray<_Tp>::operator%=(const _Tp &__t) [inline]`

Set each element *e* of array to *e* % *t*.

Definition at line 1103 of file valarray.

3.56.2.36 `template<class _Tp> valarray<_Tp> & std::valarray<_Tp>::operator%=(const valarray<_Tp> &__v) [inline]`

Modulo elements of array by corresponding elements of *v*.

Definition at line 1103 of file valarray.

3.56.2.37 `template<typename _Tp> void std::gslice_array<_Tp>::operator&= (const valarray<_Tp> &__v) const`
`[inline]`

Logical and slice elements with corresponding elements of *v*.

Definition at line 206 of file `gslice_array.h`.

3.56.2.38 `template<typename _Tp> void std::mask_array<_Tp>::operator&= (const valarray<_Tp> &__v) const`
`[inline]`

Logical and slice elements with corresponding elements of *v*.

Definition at line 196 of file `mask_array.h`.

3.56.2.39 `template<typename _Tp> void std::indirect_array<_Tp>::operator&= (const valarray<_Tp> &__v) const`
`[inline]`

Logical and slice elements with corresponding elements of *v*.

Definition at line 200 of file `indirect_array.h`.

3.56.2.40 `template<typename _Tp> void std::slice_array<_Tp>::operator&= (const valarray<_Tp> &__v) const`
`[inline]`

Logical and slice elements with corresponding elements of *v*.

Definition at line 262 of file `slice_array.h`.

3.56.2.41 `template<class _Tp> valarray<_Tp> & std::valarray<_Tp>::operator&= (const _Tp &__t)` `[inline]`

Set each element *e* of array to *e* & *t*.

Definition at line 1105 of file `valarray`.

3.56.2.42 `template<class _Tp> valarray<_Tp> & std::valarray<_Tp>::operator&= (const valarray<_Tp> &__v)`
`[inline]`

Logical and corresponding elements of *v* with elements of array.

Definition at line 1105 of file `valarray`.

3.56.2.43 `template<typename _Tp> void std::gslice_array<_Tp>::operator*=(const valarray<_Tp> &__v) const`
`[inline]`

Multiply slice elements by corresponding elements of *v*.

Definition at line 200 of file `gslice_array.h`.

3.56.2.44 `template<typename _Tp> void std::mask_array<_Tp>::operator*= (const valarray<_Tp> &__v) const`
`[inline]`

Multiply slice elements by corresponding elements of *v*.

Definition at line 190 of file mask_array.h.

3.56.2.45 `template<typename _Tp> void std::indirect_array<_Tp>::operator*= (const valarray<_Tp> &__v) const`
`[inline]`

Multiply slice elements by corresponding elements of *v*.

Definition at line 194 of file indirect_array.h.

3.56.2.46 `template<typename _Tp> void std::slice_array<_Tp>::operator*= (const valarray<_Tp> &__v) const`
`[inline]`

Multiply slice elements by corresponding elements of *v*.

Definition at line 256 of file slice_array.h.

3.56.2.47 `template<class _Tp> valarray<_Tp> & std::valarray<_Tp>::operator*= (const _Tp &__t) [inline]`

Multiply each element of array by *t*.

Definition at line 1101 of file valarray.

3.56.2.48 `template<class _Tp> valarray<_Tp> & std::valarray<_Tp>::operator*= (const valarray<_Tp> &__v)`
`[inline]`

Multiply elements of array by corresponding elements of *v*.

Definition at line 1101 of file valarray.

3.56.2.49 `template<typename _Tp> valarray<_Tp>::template _UnaryOp<__unary_plus>::_Rt std::valarray<_Tp>`
`>::operator+ () const [inline]`

Return a new valarray by applying unary + to each element.

Definition at line 1073 of file valarray.

3.56.2.50 `template<typename _Tp> void std::gslice_array<_Tp>::operator+= (const valarray<_Tp> &__v) const`
`[inline]`

Add corresponding elements of *v* to slice elements.

Definition at line 203 of file gslice_array.h.

3.56.2.51 `template<typename _Tp> void std::mask_array<_Tp>::operator+=(const valarray<_Tp> &__v) const`
`[inline]`

Add corresponding elements of *v* to slice elements.

Definition at line 193 of file `mask_array.h`.

3.56.2.52 `template<typename _Tp> void std::indirect_array<_Tp>::operator+=(const valarray<_Tp> &__v) const`
`[inline]`

Add corresponding elements of *v* to slice elements.

Definition at line 197 of file `indirect_array.h`.

3.56.2.53 `template<typename _Tp> void std::slice_array<_Tp>::operator+=(const valarray<_Tp> &__v) const`
`[inline]`

Add corresponding elements of *v* to slice elements.

Definition at line 259 of file `slice_array.h`.

3.56.2.54 `template<class _Tp> valarray<_Tp> & std::valarray<_Tp>::operator+=(const _Tp &__t)` `[inline]`

Add *t* to each element of array.

Definition at line 1099 of file `valarray`.

3.56.2.55 `template<class _Tp> valarray<_Tp> & std::valarray<_Tp>::operator+=(const valarray<_Tp> &__v)`
`[inline]`

Add corresponding elements of *v* to elements of array.

Definition at line 1099 of file `valarray`.

3.56.2.56 `template<typename _Tp> valarray<_Tp>::template _UnaryOp<__negate>::_Rt std::valarray<_Tp>::operator-`
`() const` `[inline]`

Return a new valarray by applying unary - to each element.

Definition at line 1074 of file `valarray`.

3.56.2.57 `template<typename _Tp> void std::gslice_array<_Tp>::operator-= (const valarray<_Tp> &__v) const`
`[inline]`

Subtract corresponding elements of *v* from slice elements.

Definition at line 204 of file `gslice_array.h`.

3.56.2.58 `template<typename _Tp> void std::mask_array<_Tp>::operator-= (const valarray<_Tp> &__v) const`
`[inline]`

Subtract corresponding elements of *v* from slice elements.

Definition at line 194 of file `mask_array.h`.

3.56.2.59 `template<typename _Tp> void std::indirect_array<_Tp>::operator-= (const valarray<_Tp> &__v) const`
`[inline]`

Subtract corresponding elements of *v* from slice elements.

Definition at line 198 of file `indirect_array.h`.

3.56.2.60 `template<typename _Tp> void std::slice_array<_Tp>::operator-= (const valarray<_Tp> &__v) const`
`[inline]`

Subtract corresponding elements of *v* from slice elements.

Definition at line 260 of file `slice_array.h`.

3.56.2.61 `template<class _Tp> valarray<_Tp> &std::valarray<_Tp>::operator-= (const _Tp &__t)` `[inline]`

Subtract *t* to each element of array.

Definition at line 1100 of file `valarray`.

3.56.2.62 `template<class _Tp> valarray<_Tp> &std::valarray<_Tp>::operator-= (const valarray<_Tp> &__v)`
`[inline]`

Subtract corresponding elements of *v* from elements of array.

Definition at line 1100 of file `valarray`.

3.56.2.63 `template<typename _Tp> void std::gslice_array<_Tp>::operator/= (const valarray<_Tp> &__v) const`
`[inline]`

Divide slice elements by corresponding elements of *v*.

Definition at line 201 of file `gslice_array.h`.

3.56.2.64 `template<typename _Tp> void std::mask_array<_Tp>::operator/= (const valarray<_Tp> &__v) const`
`[inline]`

Divide slice elements by corresponding elements of *v*.

Definition at line 191 of file `mask_array.h`.

3.56.2.65 `template<typename _Tp> void std::indirect_array<_Tp>::operator/= (const valarray<_Tp> &__v) const`
`[inline]`

Divide slice elements by corresponding elements of *v*.

Definition at line 195 of file `indirect_array.h`.

3.56.2.66 `template<typename _Tp> void std::slice_array<_Tp>::operator/= (const valarray<_Tp> &__v) const`
`[inline]`

Divide slice elements by corresponding elements of *v*.

Definition at line 257 of file `slice_array.h`.

3.56.2.67 `template<class _Tp> valarray<_Tp> & std::valarray<_Tp>::operator/= (const _Tp &__t)` `[inline]`

Divide each element of array by *t*.

Definition at line 1102 of file `valarray`.

3.56.2.68 `template<class _Tp> valarray<_Tp> & std::valarray<_Tp>::operator/= (const valarray<_Tp> &__v)`
`[inline]`

Divide elements of array by corresponding elements of *v*.

Definition at line 1102 of file `valarray`.

3.56.2.69 `template<typename _Tp> void std::gslice_array<_Tp>::operator<<= (const valarray<_Tp> &__v) const`
`[inline]`

Left shift slice elements by corresponding elements of *v*.

Definition at line 208 of file `gslice_array.h`.

3.56.2.70 `template<typename _Tp> void std::mask_array<_Tp>::operator<<= (const valarray<_Tp> &__v) const`
`[inline]`

Left shift slice elements by corresponding elements of *v*.

Definition at line 198 of file `mask_array.h`.

3.56.2.71 `template<typename _Tp> void std::indirect_array<_Tp>::operator<<= (const valarray<_Tp> &__v) const`
`[inline]`

Left shift slice elements by corresponding elements of *v*.

Definition at line 202 of file `indirect_array.h`.

3.56.2.72 `template<typename _Tp> void std::slice_array<_Tp>::operator<<= (const valarray<_Tp> &__v) const [inline]`

Left shift slice elements by corresponding elements of *v*.

Definition at line 264 of file `slice_array.h`.

3.56.2.73 `template<class _Tp> valarray<_Tp> & std::valarray<_Tp>::operator<<= (const _Tp &__t) [inline]`

Left shift each element *e* of array by *t* bits.

Definition at line 1107 of file `valarray`.

3.56.2.74 `template<class _Tp> valarray<_Tp> & std::valarray<_Tp>::operator<<= (const valarray<_Tp> &__v) [inline]`

Left shift elements of array by corresponding elements of *v*.

Definition at line 1107 of file `valarray`.

3.56.2.75 `template<typename _Tp> gslice_array<_Tp> & std::gslice_array<_Tp>::operator= (const gslice_array<_Tp> &__a) [inline]`

Assignment operator. Assigns slice elements to corresponding elements of *a*.

Definition at line 148 of file `gslice_array.h`.

References `std::valarray<_Tp>::size()`.

Referenced by `std::gslice_array<_Tp>::operator=()`.

3.56.2.76 `template<typename _Tp> indirect_array<_Tp> & std::indirect_array<_Tp>::operator= (const indirect_array<_Tp> &__a) [inline]`

Assignment operator. Assigns elements to corresponding elements of *a*.

Definition at line 154 of file `indirect_array.h`.

Referenced by `std::indirect_array<_Tp>::operator=()`.

3.56.2.77 `template<typename _Tp> mask_array<_Tp> & std::mask_array<_Tp>::operator= (const mask_array<_Tp> &__a) [inline]`

Assignment operator. Assigns elements to corresponding elements of *a*.

Definition at line 149 of file `mask_array.h`.

Referenced by `std::mask_array<_Tp>::operator=()`.

3.56.2.78 `template<typename _Tp> void std::gslice_array<_Tp>::operator= (const valarray<_Tp> &__v) const`
`[inline]`

Assign slice elements to corresponding elements of *v*.

Definition at line 166 of file `gslice_array.h`.

References `std::gslice_array<_Tp>::operator=()`, and `std::valarray<_Tp>::size()`.

3.56.2.79 `template<typename _Tp> void std::indirect_array<_Tp>::operator= (const valarray<_Tp> &__v) const`
`[inline]`

Assign slice elements to corresponding elements of *v*.

Definition at line 168 of file `indirect_array.h`.

References `std::indirect_array<_Tp>::operator=()`.

3.56.2.80 `gslice & std::gslice::operator= (const gslice &__g)` `[inline]`

Assignment operator.

Definition at line 170 of file `gslice.h`.

3.56.2.81 `template<typename _Tp> void std::gslice_array<_Tp>::operator= (const _Tp &__t) const` `[inline]`

Assign all slice elements to *t*.

Definition at line 158 of file `gslice_array.h`.

References `std::valarray<_Tp>::size()`.

3.56.2.82 `template<typename _Tp> void std::mask_array<_Tp>::operator= (const _Tp &__t) const` `[inline]`

Assign all slice elements to *t*.

Definition at line 158 of file `mask_array.h`.

References `std::mask_array<_Tp>::operator=()`, and `std::valarray<_Tp>::size()`.

3.56.2.83 `template<typename _Tp> void std::indirect_array<_Tp>::operator= (const _Tp &__t) const` `[inline]`

Assign all slice elements to *t*.

Definition at line 163 of file `indirect_array.h`.

3.56.2.84 `template<typename _Tp> slice_array<_Tp> & std::slice_array<_Tp>::operator= (const slice_array<_Tp> &__a) [inline]`

Assignment operator. Assigns slice elements to corresponding elements of *a*.

Definition at line 215 of file slice_array.h.

Referenced by `std::slice_array<_Tp>::operator=()`.

3.56.2.85 `template<typename _Tp> void std::slice_array<_Tp>::operator= (const valarray<_Tp> &__v) const [inline]`

Assign slice elements to corresponding elements of *v*.

Definition at line 229 of file slice_array.h.

References `std::slice_array<_Tp>::operator=()`.

3.56.2.86 `template<typename _Tp> void std::slice_array<_Tp>::operator= (const _Tp &__t) const [inline]`

Assign all slice elements to *t*.

Definition at line 224 of file slice_array.h.

3.56.2.87 `template<typename _Tp> valarray<_Tp> & std::valarray<_Tp>::operator= (const valarray<_Tp> &__v) [inline]`

Assign elements to an array.

Assign elements of array to values in *v*.

Parameters

<code>__v</code>	Valarray to get values from.
------------------	------------------------------

Definition at line 708 of file valarray.

3.56.2.88 `template<typename _Tp> valarray<_Tp> & std::valarray<_Tp>::operator= (valarray<_Tp> &&__v) [inline], [noexcept]`

Move assign elements to an array.

Move assign elements of array to values in *v*.

Parameters

<code>__v</code>	Valarray to get values from.
------------------	------------------------------

Definition at line 732 of file valarray.

3.56.2.89 `template<typename _Tp> valarray<_Tp> & std::valarray<_Tp>::operator= (const _Tp & __t) [inline]`

Assign elements to a value.

Assign all elements of array to *t*.

Parameters

↔	Value for elements.
↔	
↔	
↔	
<i>t</i>	

Definition at line 772 of file valarray.

3.56.2.90 `template<typename _Tp> valarray<_Tp> & std::valarray<_Tp>::operator= (const slice_array<_Tp> & __sa) [inline]`

Assign elements to an array subset.

Assign elements of array to values in *sa*. Results are undefined if *sa* does not have the same size as this array.

Parameters

<i>__sa</i>	Array slice to get values from.
-------------	---------------------------------

Definition at line 780 of file valarray.

3.56.2.91 `template<typename _Tp> valarray<_Tp> & std::valarray<_Tp>::operator= (const gslice_array<_Tp> & __ga) [inline]`

Assign elements to an array subset.

Assign elements of array to values in *ga*. Results are undefined if *ga* does not have the same size as this array.

Parameters

<i>__ga</i>	Array slice to get values from.
-------------	---------------------------------

Definition at line 790 of file valarray.

3.56.2.92 `template<typename _Tp> valarray<_Tp> & std::valarray<_Tp>::operator= (const mask_array<_Tp> & __ma) [inline]`

Assign elements to an array subset.

Assign elements of array to values in *ma*. Results are undefined if *ma* does not have the same size as this array.

Parameters

<code>__ma</code>	Array slice to get values from.
-------------------	---------------------------------

Definition at line 800 of file `valarray`.

3.56.2.93 `template<typename _Tp> valarray<_Tp> & std::valarray<_Tp>::operator= (const indirect_array<_Tp> & __ia) [inline]`

Assign elements to an array subset.

Assign elements of array to values in *ia*. Results are undefined if *ia* does not have the same size as this array.

Parameters

<code>__↵ __ia</code>	Array slice to get values from.
---------------------------	---------------------------------

Definition at line 810 of file `valarray`.

3.56.2.94 `template<typename _Tp> valarray<_Tp> & std::valarray<_Tp>::operator= (initializer_list<_Tp> __l) [inline]`

Assign elements to an `initializer_list`.

Assign elements of array to values in `__l`. Results are undefined if `__l` does not have the same size as this array.

Parameters

<code>↵ __↵ ↵ __↵ l</code>	<code>initializer_list</code> to get values from.
--	---

Definition at line 748 of file `valarray`.

3.56.2.95 `template<typename _Tp> void std::gslice_array<_Tp>::operator>=> (const valarray<_Tp> & __v) const [inline]`

Right shift slice elements by corresponding elements of *v*.

Definition at line 209 of file `gslice_array.h`.

3.56.2.96 `template<typename _Tp> void std::mask_array<_Tp>::operator>>= (const valarray<_Tp> &__v) const`
`[inline]`

Right shift slice elements by corresponding elements of *v*.

Definition at line 199 of file `mask_array.h`.

3.56.2.97 `template<typename _Tp> void std::indirect_array<_Tp>::operator>>= (const valarray<_Tp> &__v) const`
`[inline]`

Right shift slice elements by corresponding elements of *v*.

Definition at line 203 of file `indirect_array.h`.

3.56.2.98 `template<typename _Tp> void std::slice_array<_Tp>::operator>>= (const valarray<_Tp> &__v) const`
`[inline]`

Right shift slice elements by corresponding elements of *v*.

Definition at line 265 of file `slice_array.h`.

3.56.2.99 `template<class _Tp> valarray<_Tp> &std::valarray<_Tp>::operator>>= (const _Tp &__t)` `[inline]`

Right shift each element *e* of array by *t* bits.

Definition at line 1108 of file `valarray`.

3.56.2.100 `template<class _Tp> valarray<_Tp> &std::valarray<_Tp>::operator>>= (const valarray<_Tp> &__v)`
`[inline]`

Right shift elements of array by corresponding elements of *v*.

Definition at line 1108 of file `valarray`.

3.56.2.101 `template<typename _Tp> _Tp &std::valarray<_Tp>::operator[] (size_t __i)` `[inline]`

Return a reference to the *i*'th array element.

Parameters

<code>↵</code>	Index of element to return.
<code>__↵</code>	
<code>↵</code>	
<code>__↵</code>	
<code><i>i</i></code>	

Returns

Reference to the i'th element.

Definition at line 576 of file valarray.

3.56.2.102 `template<typename _Tp> _Expr<_SClos<_ValArray,_Tp>,_Tp> std::valarray<_Tp>::operator[] (slice __s) const [inline]`

Return an array subset.

Returns a new valarray containing the elements of the array indicated by the slice argument. The new valarray has the same size as the input slice.

See also

slice.

Parameters

<code>__s</code>	The source slice.
------------------	-------------------

Returns

New valarray containing elements in `__s`.

Definition at line 842 of file valarray.

3.56.2.103 `template<typename _Tp> slice_array<_Tp> std::valarray<_Tp>::operator[] (slice __s) [inline]`

Return a reference to an array subset.

Returns a new valarray containing the elements of the array indicated by the slice argument. The new valarray has the same size as the input slice.

See also

slice.

Parameters

<code>__s</code>	The source slice.
------------------	-------------------

Returns

New valarray containing elements in `__s`.

Definition at line 850 of file valarray.

```
3.56.2.104 template<typename _Tp> _Expr<_GClos<_ValArray, _Tp>, _Tp> std::valarray<_Tp>::operator[] ( const
        gslice & __s ) const    [inline]
```

Return an array subset.

Returns a slice_array referencing the elements of the array indicated by the slice argument.

See also

gslice.

Parameters

<code>__↔ __s</code>	The source slice.
--------------------------	-------------------

Returns

Slice_array referencing elements indicated by `__s`.

Definition at line 855 of file valarray.

```
3.56.2.105 template<typename _Tp> gslice_array<_Tp> std::valarray<_Tp>::operator[] ( const gslice & __s )
        [inline]
```

Return a reference to an array subset.

Returns a new valarray containing the elements of the array indicated by the gslice argument. The new valarray has the same size as the input gslice.

See also

gslice.

Parameters

<code>__↔ __s</code>	The source gslice.
--------------------------	--------------------

Returns

New valarray containing elements in `__s`.

Definition at line 864 of file valarray.

3.56.2.106 `template<typename _Tp> valarray<_Tp> std::valarray<_Tp>::operator[] (const valarray<bool> & __m)
const [inline]`

Return an array subset.

Returns a new valarray containing the elements of the array indicated by the argument. The input is a valarray of bool which represents a bitmask indicating which elements should be copied into the new valarray. Each element of the array is added to the return valarray if the corresponding element of the argument is true.

Parameters

<code>__m</code>	The valarray bitmask.
------------------	-----------------------

Returns

New valarray containing elements indicated by `__m`.

Definition at line 872 of file valarray.

3.56.2.107 `template<typename _Tp> mask_array<_Tp> std::valarray<_Tp>::operator[] (const valarray<bool> & __m) [inline]`

Return a reference to an array subset.

Returns a new mask_array referencing the elements of the array indicated by the argument. The input is a valarray of bool which represents a bitmask indicating which elements are part of the subset. Elements of the array are part of the subset if the corresponding element of the argument is true.

Parameters

<code>__m</code>	The valarray bitmask.
------------------	-----------------------

Returns

New valarray containing elements indicated by `__m`.

Definition at line 884 of file valarray.

3.56.2.108 `template<typename _Tp> _Expr<_IClos<_ValArray, _Tp>, _Tp> std::valarray<_Tp>::operator[] (const valarray<size_t> &__i) const [inline]`

Return an array subset.

Returns a new valarray containing the elements of the array indicated by the argument. The elements in the argument are interpreted as the indices of elements of this valarray to copy to the return valarray.

Parameters

↩	The valarray element index list.
__↩	
↩	
__↩	
<i>i</i>	

Returns

New valarray containing elements in __s.

Definition at line 895 of file valarray.

3.56.2.109 `template<typename _Tp> indirect_array<_Tp> std::valarray<_Tp>::operator[] (const valarray<size_t> &__i) [inline]`

Return a reference to an array subset.

Returns an indirect_array referencing the elements of the array indicated by the argument. The elements in the argument are interpreted as the indices of elements of this valarray to include in the subset. The returned indirect_array refers to these elements.

Parameters

↩	The valarray element index list.
__↩	
↩	
__↩	
<i>i</i>	

Returns

Indirect_array referencing elements in __i.

Definition at line 903 of file valarray.

3.56.2.110 `template<typename _Tp> void std::gslice_array<_Tp>::operator^= (const valarray<_Tp> &__v) const [inline]`

Logical xor slice elements with corresponding elements of v.

Definition at line 205 of file gslice_array.h.

3.56.2.111 `template<typename _Tp> void std::mask_array<_Tp>::operator^= (const valarray<_Tp> &__v) const`
`[inline]`

Logical xor slice elements with corresponding elements of v .

Definition at line 195 of file mask_array.h.

3.56.2.112 `template<typename _Tp> void std::indirect_array<_Tp>::operator^= (const valarray<_Tp> &__v) const`
`[inline]`

Logical xor slice elements with corresponding elements of v .

Definition at line 199 of file indirect_array.h.

3.56.2.113 `template<typename _Tp> void std::slice_array<_Tp>::operator^= (const valarray<_Tp> &__v) const`
`[inline]`

Logical xor slice elements with corresponding elements of v .

Definition at line 261 of file slice_array.h.

3.56.2.114 `template<class _Tp> valarray<_Tp> &std::valarray<_Tp>::operator^= (const _Tp &__t) [inline]`

Set each element e of array to $e \wedge t$.

Definition at line 1104 of file valarray.

3.56.2.115 `template<class _Tp> valarray<_Tp> &std::valarray<_Tp>::operator^= (const valarray<_Tp> &__v)`
`[inline]`

Logical xor corresponding elements of v with elements of array.

Definition at line 1104 of file valarray.

3.56.2.116 `template<typename _Tp> void std::gslice_array<_Tp>::operator|= (const valarray<_Tp> &__v) const`
`[inline]`

Logical or slice elements with corresponding elements of v .

Definition at line 207 of file gslice_array.h.

3.56.2.117 `template<typename _Tp> void std::mask_array<_Tp>::operator|= (const valarray<_Tp> &__v) const`
`[inline]`

Logical or slice elements with corresponding elements of v .

Definition at line 197 of file mask_array.h.

3.56.2.118 `template<typename _Tp> void std::indirect_array<_Tp>::operator|= (const valarray<_Tp> &__v) const`
`[inline]`

Logical or slice elements with corresponding elements of *v*.

Definition at line 201 of file `indirect_array.h`.

3.56.2.119 `template<typename _Tp> void std::slice_array<_Tp>::operator|= (const valarray<_Tp> &__v) const`
`[inline]`

Logical or slice elements with corresponding elements of *v*.

Definition at line 263 of file `slice_array.h`.

3.56.2.120 `template<class _Tp> valarray<_Tp> &std::valarray<_Tp>::operator|= (const _Tp &__t)` `[inline]`

Set each element *e* of array to *e* | *t*.

Definition at line 1106 of file `valarray`.

3.56.2.121 `template<class _Tp> valarray<_Tp> &std::valarray<_Tp>::operator|= (const valarray<_Tp> &__v)`
`[inline]`

Logical or corresponding elements of *v* with elements of array.

Definition at line 1106 of file `valarray`.

3.56.2.122 `template<typename _Tp> valarray<_Tp>::template _UnaryOp<__bitwise_not>::Rt std::valarray<_Tp>`
`>::operator~() const` `[inline]`

Return a new valarray by applying unary `~` to each element.

Definition at line 1075 of file `valarray`.

3.56.2.123 `template<class _Tp> void std::valarray<_Tp>::resize (size_t __size, _Tp __c = _Tp())` `[inline]`

Resize array.

Resize this array to *size* and set all elements to *c*. All references and iterators are invalidated.

Parameters

<code>__size</code>	New array size.
<code>__c</code>	New value for all elements.

Definition at line 1016 of file `valarray`.

3.56.2.124 `template<class _Tp> valarray<_Tp> std::valarray<_Tp>::shift(int __n) const [inline]`

Return a shifted array.

A new valarray is constructed as a copy of this array with elements in shifted positions. For an element with index *i*, the new position is *i* - *n*. The new valarray has the same size as the current one. New elements without a value are set to 0. Elements whose new position is outside the bounds of the array are discarded.

Positive arguments shift toward index 0, discarding elements [0, *n*). Negative arguments discard elements from the top of the array.

Parameters

<code>__n</code>	Number of element positions to shift.
------------------	---------------------------------------

Returns

New valarray with elements in shifted positions.

Definition at line 934 of file valarray.

3.56.2.125 `size_t std::slice::size() const [inline]`

Return size of slice.

Definition at line 102 of file slice_array.h.

Referenced by `std::slice::stride()`.

3.56.2.126 `valarray< size_t> std::gslice::size() const [inline]`

Return array of sizes of slice dimensions.

Definition at line 139 of file gslice.h.

3.56.2.127 `template<class _Tp> size_t std::valarray<_Tp>::size() const [inline]`

Return the number of elements in array.

Definition at line 921 of file valarray.

Referenced by `std::gslice_array<_Tp>::operator=()`, and `std::mask_array<_Tp>::operator=()`.

3.56.2.128 `size_t std::slice::start() const [inline]`

Return array offset of first slice element.

Definition at line 98 of file slice_array.h.

Referenced by `std::slice::stride()`.

3.56.2.129 `size_t std::gslice::start () const [inline]`

Return array offset of first slice element.

Definition at line 135 of file gslice.h.

3.56.2.130 `size_t std::slice::stride () const [inline]`

Return array stride of slice.

Definition at line 106 of file slice_array.h.

References `std::slice::size()`, `std::slice::start()`, and `std::slice::stride()`.

Referenced by `std::slice::stride()`.

3.56.2.131 `valarray< size_t > std::gslice::stride () const [inline]`

Return array of array strides for each dimension.

Definition at line 143 of file gslice.h.

3.56.2.132 `template<class _Tp> _Tp std::valarray< _Tp >::sum () const [inline]`

Return the sum of all elements in the array.

Accumulates the sum of all elements into a `Tp` using `+=`. The order of adding the elements is unspecified.

Definition at line 926 of file valarray.

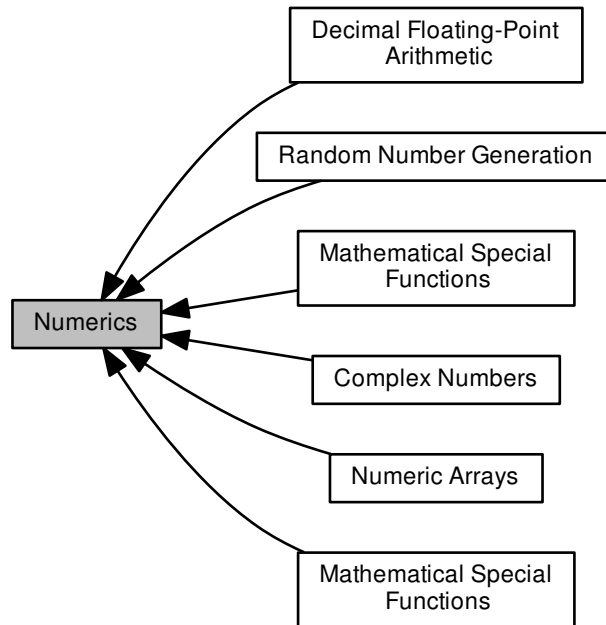
3.56.2.133 `template<class _Tp> void std::valarray< _Tp >::swap (valarray< _Tp > & __v) [inline],
[noexcept]`

Swap.

Definition at line 912 of file valarray.

3.57 Numerics

Collaboration diagram for Numerics:



Modules

- [Complex Numbers](#)
- [Decimal Floating-Point Arithmetic](#)
- [Mathematical Special Functions](#)
- [Mathematical Special Functions](#)
- [Numeric Arrays](#)
- [Random Number Generation](#)

3.57.1 Detailed Description

Components for performing numeric operations. Includes support for complex number types, random number generation, numeric (n-at-a-time) arrays, generalized numeric algorithms, and special math functions.

3.58 Optional values

Collaboration diagram for Optional values:



Classes

- struct `std::experimental::fundamentals_v1::_Has_addressof< _Tp >`
- class `std::experimental::fundamentals_v1::_Optional_base< _Tp, _ShouldProvideDestructor >`
- class `std::experimental::fundamentals_v1::_Optional_base< _Tp, false >`
- class `std::experimental::fundamentals_v1::bad_optional_access`
- struct `std::experimental::fundamentals_v1::in_place_t`
- struct `std::experimental::fundamentals_v1::nullopt_t`
- class `std::experimental::fundamentals_v1::optional< _Tp >`

Macros

- `#define __cpp_lib_experimental_optional`

Typedefs

- `template<typename _Tp, typename _Up >`
using `std::experimental::fundamentals_v1::_assigns_from_optional` = `__or_< is_assignable< _Tp &, const optional< _Up > & >, is_assignable< _Tp &, optional< _Up > & >, is_assignable< _Tp &, const optional< _Up > && >, is_assignable< _Tp &, optional< _Up > && >>`
- `template<typename _Tp, typename _Up >`
using `std::experimental::fundamentals_v1::_converts_from_optional` = `__or_< is_constructible< _Tp, const optional< _Up > & >, is_constructible< _Tp, optional< _Up > & >, is_constructible< _Tp, const optional< _Up > && >, is_constructible< _Tp, optional< _Up > && >, is_convertible< const optional< _Up > &, _Tp >, is_convertible< optional< _Up > &, _Tp >, is_convertible< const optional< _Up > &&, _Tp >, is_convertible< optional< _Up > &&, _Tp >>`

Functions

- `template<typename _Tp >`
`constexpr enable_if_t<!_Has_addressof< _Tp >::value, _Tp * > std::experimental::fundamentals_v1::__`
`constexpr_addressof (_Tp &__t)`
- `void std::experimental::fundamentals_v1::__throw_bad_optional_access (const char *) __attribute__((__`
`noreturn__))`
- `template<typename _Tp >`
`constexpr optional< decay_t< _Tp > > std::experimental::fundamentals_v1::make_optional (_Tp &&__t)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator!= (const optional< _Tp > &__lhs, const`
`optional< _Tp > &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator!= (const optional< _Tp > &__lhs, nullopt_t`
`__t) noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator!= (nullopt_t, const optional< _Tp > &__rhs)`
`noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator!= (const optional< _Tp > &__lhs, _Tp const`
`&__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator!= (const _Tp &__lhs, const optional< _Tp >`
`&__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator< (const optional< _Tp > &__lhs, const`
`optional< _Tp > &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator< (const optional< _Tp > &, nullopt_t) noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator< (nullopt_t, const optional< _Tp > &__rhs)`
`noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator< (const optional< _Tp > &__lhs, const _Tp`
`&__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator< (const _Tp &__lhs, const optional< _Tp >`
`&__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator<= (const optional< _Tp > &__lhs, const`
`optional< _Tp > &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator<= (const optional< _Tp > &__lhs, nullopt_t)`
`noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator<= (nullopt_t, const optional< _Tp > &) noex-`
`cept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator<= (const optional< _Tp > &__lhs, const _Tp`
`&__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator<= (const _Tp &__lhs, const optional< _Tp >`
`&__rhs)`

- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator== (const optional< _Tp > &__lhs, const optional< _Tp > &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator== (const optional< _Tp > &__lhs, nullopt_t) noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator== (nullopt_t, const optional< _Tp > &__rhs) noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator== (const optional< _Tp > &__lhs, const _Tp &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator== (const _Tp &__lhs, const optional< _Tp > &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator> (const optional< _Tp > &__lhs, const optional< _Tp > &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator> (const optional< _Tp > &__lhs, nullopt_t) noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator> (nullopt_t, const optional< _Tp > &) noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator> (const optional< _Tp > &__lhs, const _Tp &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator> (const _Tp &__lhs, const optional< _Tp > &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator>= (const optional< _Tp > &__lhs, const optional< _Tp > &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator>= (const optional< _Tp > &, nullopt_t) noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator>= (nullopt_t, const optional< _Tp > &__rhs) noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator>= (const optional< _Tp > &__lhs, const _Tp &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator>= (const _Tp &__lhs, const optional< _Tp > &__rhs)`
- `template<typename _Tp >`
`void std::experimental::fundamentals_v1::swap (optional< _Tp > &__lhs, optional< _Tp > &__rhs) noexcept(noexcept(__lhs.swap(__rhs)))`

Variables

- `constexpr in_place_t std::experimental::fundamentals_v1::in_place`
- `constexpr nullopt_t std::experimental::fundamentals_v1::nullopt`

3.58.1 Detailed Description

Class template for optional values and surrounding facilities, as described in n3793 "A proposal to add a utility class to represent optional objects (Revision 5)".

3.58.2 Function Documentation

3.58.2.1 `template<typename _Tp > constexpr enable_if_t<!Has_addressof<_Tp>::value, _Tp*>
std::experimental::fundamentals_v1::__constexpr_addressof (_Tp & __t)`

An overload that attempts to take the address of an lvalue as a constant expression. Falls back to `__addressof` in the presence of an overloaded `addressof` operator (unary operator`&`), in which case the call will not be a constant expression.

Definition at line 177 of file `optional`.

3.58.3 Variable Documentation

3.58.3.1 `constexpr in_place_t std::experimental::fundamentals_v1::in_place`

Tag for in-place construction.

Definition at line 90 of file `optional`.

3.58.3.2 `constexpr nullopt_t std::experimental::fundamentals_v1::nullopt`

Tag to disengage optional objects.

Definition at line 109 of file `optional`.

3.59 Pointer Abstractions

Collaboration diagram for Pointer Abstractions:



Classes

- struct `std::default_delete<_Tp>`
- struct `std::default_delete<_Tp[]>`
- class `std::enable_shared_from_this<_Tp>`
- struct `std::hash<shared_ptr<_Tp>>`
- struct `std::hash<unique_ptr<_Tp, _Dp>>`
- struct `std::owner_less<_Tp>`
- struct `std::owner_less<shared_ptr<_Tp>>`
- struct `std::owner_less<weak_ptr<_Tp>>`
- struct `std::pointer_traits<_Ptr>`
- struct `std::pointer_traits<_Tp*>`
- class `std::shared_ptr<_Tp>`
- class `std::unique_ptr<_Tp, _Dp>`
- class `std::unique_ptr<_Tp[], _Dp>`
- class `std::weak_ptr<_Tp>`

Macros

- `#define __cpp_lib_make_unique`

Functions

- `template<typename _Tp1, typename _Tp2>`
`void std::enable_shared_from_this_helper (const __shared_count<> &__pn, const enable_shared_from_↵`
`_this<_Tp1> *__pe, const _Tp2 *__px) noexcept`
- `template<typename _Tp, typename _Alloc, typename... _Args>`
`shared_ptr<_Tp> std::allocate_shared (const _Alloc &__a, _Args &&...__args)`
- `template<typename _Tp, typename _Tp1>`
`shared_ptr<_Tp> std::const_pointer_cast (const shared_ptr<_Tp1> &__r) noexcept`
- `template<typename _Tp, typename _Tp1>`
`shared_ptr<_Tp> std::dynamic_pointer_cast (const shared_ptr<_Tp1> &__r) noexcept`
- `template<typename _Del, typename _Tp, _Lock_policy _Lp>`
`_Del * std::get_deleter (const __shared_ptr<_Tp, _Lp> &__p) noexcept`

- `template<typename _Tp, typename... _Args>`
`shared_ptr< _Tp > std::make_shared (_Args &&...__args)`
- `template<typename _Tp, typename... _Args>`
`_MakeUniq< _Tp >::__single_object std::make_unique (_Args &&...__args)`
- `template<typename _Tp >`
`_MakeUniq< _Tp >::__array std::make_unique (size_t __num)`
- `template<typename _Tp, typename... _Args>`
`_MakeUniq< _Tp >::__invalid_type std::make_unique (_Args &&...)=delete`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::operator!= (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::operator!= (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool std::operator!= (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`bool std::operator!= (const unique_ptr< _Tp, _Dp > &__x, const unique_ptr< _Up, _Ep > &__y)`
- `template<typename _Tp, typename _Dp >`
`bool std::operator!= (const unique_ptr< _Tp, _Dp > &__x, nullptr_t) noexcept`
- `template<typename _Tp, typename _Dp >`
`bool std::operator!= (nullptr_t, const unique_ptr< _Tp, _Dp > &__x) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::operator< (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::operator< (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool std::operator< (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`bool std::operator< (const unique_ptr< _Tp, _Dp > &__x, const unique_ptr< _Up, _Ep > &__y)`
- `template<typename _Tp, typename _Dp >`
`bool std::operator< (const unique_ptr< _Tp, _Dp > &__x, nullptr_t)`
- `template<typename _Tp, typename _Dp >`
`bool std::operator< (nullptr_t, const unique_ptr< _Tp, _Dp > &__x)`
- `template<typename _Ch, typename _Tr, typename _Tp, _Lock_policy _Lp>`
`std::basic_ostream< _Ch, _Tr > & std::operator<< (std::basic_ostream< _Ch, _Tr > &__os, const __shared_ptr< _Tp, _Lp > &__p)`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::operator<= (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::operator<= (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool std::operator<= (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`bool std::operator<= (const unique_ptr< _Tp, _Dp > &__x, const unique_ptr< _Up, _Ep > &__y)`
- `template<typename _Tp, typename _Dp >`
`bool std::operator<= (const unique_ptr< _Tp, _Dp > &__x, nullptr_t)`
- `template<typename _Tp, typename _Dp >`
`bool std::operator<= (nullptr_t, const unique_ptr< _Tp, _Dp > &__x)`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::operator== (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::operator== (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool std::operator== (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`

- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`bool std::operator== (const unique_ptr< _Tp, _Dp > &__x, const unique_ptr< _Up, _Ep > &__y)`
- `template<typename _Tp, typename _Dp >`
`bool std::operator== (const unique_ptr< _Tp, _Dp > &__x, nullptr_t) noexcept`
- `template<typename _Tp, typename _Dp >`
`bool std::operator== (nullptr_t, const unique_ptr< _Tp, _Dp > &__x) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::operator> (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::operator> (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool std::operator> (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`bool std::operator> (const unique_ptr< _Tp, _Dp > &__x, const unique_ptr< _Up, _Ep > &__y)`
- `template<typename _Tp, typename _Dp >`
`bool std::operator> (const unique_ptr< _Tp, _Dp > &__x, nullptr_t)`
- `template<typename _Tp, typename _Dp >`
`bool std::operator> (nullptr_t, const unique_ptr< _Tp, _Dp > &__x)`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::operator>= (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::operator>= (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool std::operator>= (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`bool std::operator>= (const unique_ptr< _Tp, _Dp > &__x, const unique_ptr< _Up, _Ep > &__y)`
- `template<typename _Tp, typename _Dp >`
`bool std::operator>= (const unique_ptr< _Tp, _Dp > &__x, nullptr_t)`
- `template<typename _Tp, typename _Dp >`
`bool std::operator>= (nullptr_t, const unique_ptr< _Tp, _Dp > &__x)`
- `template<typename _Tp, typename _Tp1 >`
`shared_ptr< _Tp > std::static_pointer_cast (const shared_ptr< _Tp1 > &__r) noexcept`
- `template<typename _Tp >`
`void std::swap (shared_ptr< _Tp > &__a, shared_ptr< _Tp > &__b) noexcept`
- `template<typename _Tp >`
`void std::swap (weak_ptr< _Tp > &__a, weak_ptr< _Tp > &__b) noexcept`
- `template<typename _Tp, typename _Dp >`
`void std::swap (unique_ptr< _Tp, _Dp > &__x, unique_ptr< _Tp, _Dp > &__y) noexcept`

- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::atomic_is_lock_free (const __shared_ptr< _Tp, _Lp > *__p)`
- `template<typename _Tp >`
`bool std::atomic_is_lock_free (const shared_ptr< _Tp > *__p)`

- `template<typename _Tp >`
`shared_ptr< _Tp > std::atomic_load_explicit (const shared_ptr< _Tp > *__p, memory_order)`
- `template<typename _Tp >`
`shared_ptr< _Tp > std::atomic_load (const shared_ptr< _Tp > *__p)`
- `template<typename _Tp, _Lock_policy _Lp>`
`__shared_ptr< _Tp, _Lp > std::atomic_load_explicit (const __shared_ptr< _Tp, _Lp > *__p, memory_order)`

- `template<typename _Tp, _Lock_policy _Lp>`
`__shared_ptr< _Tp, _Lp > std::atomic_load (const __shared_ptr< _Tp, _Lp > *__p)`

- `template<typename _Tp >`
`void std::atomic_store_explicit (shared_ptr< _Tp > *__p, shared_ptr< _Tp > __r, memory_order)`
- `template<typename _Tp >`
`void std::atomic_store (shared_ptr< _Tp > *__p, shared_ptr< _Tp > __r)`
- `template<typename _Tp, _Lock_policy _Lp>`
`void std::atomic_store_explicit (__shared_ptr< _Tp, _Lp > *__p, __shared_ptr< _Tp, _Lp > __r, memory_order)`
- `template<typename _Tp, _Lock_policy _Lp>`
`void std::atomic_store (__shared_ptr< _Tp, _Lp > *__p, __shared_ptr< _Tp, _Lp > __r)`

- `template<typename _Tp >`
`shared_ptr< _Tp > std::atomic_exchange_explicit (shared_ptr< _Tp > *__p, shared_ptr< _Tp > __r, memory_order)`
- `template<typename _Tp >`
`shared_ptr< _Tp > std::atomic_exchange (shared_ptr< _Tp > *__p, shared_ptr< _Tp > __r)`
- `template<typename _Tp, _Lock_policy _Lp>`
`__shared_ptr< _Tp, _Lp > std::atomic_exchange_explicit (__shared_ptr< _Tp, _Lp > *__p, __shared_ptr< _Tp, _Lp > __r, memory_order)`
- `template<typename _Tp, _Lock_policy _Lp>`
`__shared_ptr< _Tp, _Lp > std::atomic_exchange (__shared_ptr< _Tp, _Lp > *__p, __shared_ptr< _Tp, _Lp > __r)`

- `template<typename _Tp >`
`bool std::atomic_compare_exchange_strong_explicit (shared_ptr< _Tp > *__p, shared_ptr< _Tp > *__v, shared_ptr< _Tp > __w, memory_order, memory_order)`
- `template<typename _Tp >`
`bool std::atomic_compare_exchange_strong (shared_ptr< _Tp > *__p, shared_ptr< _Tp > *__v, shared_ptr< _Tp > __w)`
- `template<typename _Tp >`
`bool std::atomic_compare_exchange_weak_explicit (shared_ptr< _Tp > *__p, shared_ptr< _Tp > *__v, shared_ptr< _Tp > __w, memory_order __success, memory_order __failure)`
- `template<typename _Tp >`
`bool std::atomic_compare_exchange_weak (shared_ptr< _Tp > *__p, shared_ptr< _Tp > *__v, shared_ptr< _Tp > __w)`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::atomic_compare_exchange_strong_explicit (__shared_ptr< _Tp, _Lp > *__p, __shared_ptr< _Tp, _Lp > *__v, __shared_ptr< _Tp, _Lp > __w, memory_order, memory_order)`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::atomic_compare_exchange_strong (__shared_ptr< _Tp, _Lp > *__p, __shared_ptr< _Tp, _Lp > *__v, __shared_ptr< _Tp, _Lp > __w)`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::atomic_compare_exchange_weak_explicit (__shared_ptr< _Tp, _Lp > *__p, __shared_ptr< _Tp, _Lp > *__v, __shared_ptr< _Tp, _Lp > __w, memory_order __success, memory_order __failure)`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::atomic_compare_exchange_weak (__shared_ptr< _Tp, _Lp > *__p, __shared_ptr< _Tp, _Lp > *__v, __shared_ptr< _Tp, _Lp > __w)`

3.59.1 Detailed Description

Smart pointers, etc.

3.59.2 Function Documentation

3.59.2.1 `template<typename _Tp, typename _Alloc, typename... _Args> shared_ptr<_Tp> std::allocate_shared (const _Alloc & __a, _Args &&... __args) [inline]`

Create an object that is owned by a `shared_ptr`.

Parameters

<code>__a</code>	An allocator.
<code>__args</code>	Arguments for the <code>_Tp</code> object's constructor.

Returns

A `shared_ptr` that owns the newly created object.

Exceptions

<i>An</i>	exception thrown from <code>_Alloc::allocate</code> or from the constructor of <code>_Tp</code> .
-----------	---

A copy of `__a` will be used to allocate memory for the `shared_ptr` and the new object.

Definition at line 617 of file `bits/shared_ptr.h`.

3.59.2.2 `template<typename _Tp> bool std::atomic_compare_exchange_strong (shared_ptr<_Tp> * __p, shared_ptr<_Tp> * __v, shared_ptr<_Tp> __w) [inline]`

Atomic compare-and-swap for `shared_ptr` objects.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__v</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__w</code>	A non-null pointer to a <code>shared_ptr</code> object.

Returns

True if `*__p` was equivalent to `*__v`, false otherwise.

The memory order for failure shall not be `memory_order_release` or `memory_order_acq_rel`, or stronger than the memory order for success.

Definition at line 242 of file `shared_ptr_atomic.h`.

References `std::atomic_compare_exchange_strong_explicit()`.

3.59.2.3 `template<typename _Tp, _Lock_policy _Lp> bool std::atomic_compare_exchange_strong(__shared_ptr<_Tp, _Lp> * __p, __shared_ptr<_Tp, _Lp> * __v, __shared_ptr<_Tp, _Lp> __w) [inline]`

Atomic compare-and-swap for `shared_ptr` objects.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__v</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__w</code>	A non-null pointer to a <code>shared_ptr</code> object.

Returns

True if `*__p` was equivalent to `*__v`, false otherwise.

The memory order for failure shall not be `memory_order_release` or `memory_order_acq_rel`, or stronger than the memory order for success.

Definition at line 294 of file `shared_ptr_atomic.h`.

References `std::atomic_compare_exchange_strong_explicit()`.

3.59.2.4 `template<typename _Tp> bool std::atomic_compare_exchange_strong_explicit(shared_ptr<_Tp> * __p, shared_ptr<_Tp> * __v, shared_ptr<_Tp> __w, memory_order, memory_order)`

Atomic compare-and-swap for `shared_ptr` objects.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__v</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__w</code>	A non-null pointer to a <code>shared_ptr</code> object.

Returns

True if `*__p` was equivalent to `*__v`, false otherwise.

The memory order for failure shall not be `memory_order_release` or `memory_order_acq_rel`, or stronger than the memory order for success.

Definition at line 220 of file `shared_ptr_atomic.h`.

3.59.2.5 `template<typename _Tp, _Lock_policy _Lp> bool std::atomic_compare_exchange_strong_explicit (__shared_ptr< _Tp, _Lp > * __p, __shared_ptr< _Tp, _Lp > * __v, __shared_ptr< _Tp, _Lp > __w, memory_order, memory_order)`

Atomic compare-and-swap for `shared_ptr` objects.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__v</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__w</code>	A non-null pointer to a <code>shared_ptr</code> object.

Returns

True if `*__p` was equivalent to `*__v`, false otherwise.

The memory order for failure shall not be `memory_order_release` or `memory_order_acq_rel`, or stronger than the memory order for success.

Definition at line 272 of file `shared_ptr_atomic.h`.

Referenced by `std::atomic_compare_exchange_strong()`, and `std::atomic_compare_exchange_weak_explicit()`.

3.59.2.6 `template<typename _Tp> bool std::atomic_compare_exchange_weak (shared_ptr< _Tp > * __p, shared_ptr< _Tp > * __v, shared_ptr< _Tp > __w) [inline]`

Atomic compare-and-swap for `shared_ptr` objects.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__v</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__w</code>	A non-null pointer to a <code>shared_ptr</code> object.

Returns

True if `*__p` was equivalent to `*__v`, false otherwise.

The memory order for failure shall not be `memory_order_release` or `memory_order_acq_rel`, or stronger than the memory order for success.

Definition at line 263 of file `shared_ptr_atomic.h`.

References `std::atomic_compare_exchange_weak_explicit()`.

3.59.2.7 `template<typename _Tp, _Lock_policy _Lp> bool std::atomic_compare_exchange_weak (__shared_ptr<_Tp, _Lp> * __p, __shared_ptr<_Tp, _Lp> * __v, __shared_ptr<_Tp, _Lp> __w) [inline]`

Atomic compare-and-swap for `shared_ptr` objects.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__v</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__w</code>	A non-null pointer to a <code>shared_ptr</code> object.

Returns

True if `*__p` was equivalent to `*__v`, false otherwise.

The memory order for failure shall not be `memory_order_release` or `memory_order_acq_rel`, or stronger than the memory order for success.

Definition at line 316 of file `shared_ptr_atomic.h`.

References `std::atomic_compare_exchange_weak_explicit()`.

3.59.2.8 `template<typename _Tp> bool std::atomic_compare_exchange_weak_explicit (shared_ptr<_Tp> * __p, shared_ptr<_Tp> * __v, shared_ptr<_Tp> __w, memory_order __success, memory_order __failure) [inline]`

Atomic compare-and-swap for `shared_ptr` objects.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__v</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__w</code>	A non-null pointer to a <code>shared_ptr</code> object.

Returns

True if `*__p` was equivalent to `*__v`, false otherwise.

The memory order for failure shall not be `memory_order_release` or `memory_order_acq_rel`, or stronger than the memory order for success.

Definition at line 251 of file `shared_ptr_atomic.h`.

References `std::atomic_compare_exchange_strong_explicit()`.

```
3.59.2.9  template<typename _Tp, _Lock_policy _Lp> bool std::atomic_compare_exchange_weak_explicit ( __shared_ptr<
    _Tp, _Lp> * __p, __shared_ptr< _Tp, _Lp> * __v, __shared_ptr< _Tp, _Lp> __w, memory_order __success,
    memory_order __failure ) [inline]
```

Atomic compare-and-swap for shared_ptr objects.

Parameters

\leftarrow __p	A non-null pointer to a shared_ptr object.
\leftarrow __v	A non-null pointer to a shared_ptr object.
\leftarrow __w	A non-null pointer to a shared_ptr object.

Returns

True if *__p was equivalent to *__v, false otherwise.

The memory order for failure shall not be memory_order_release or memory_order_acq_rel, or stronger than the memory order for success.

Definition at line 304 of file shared_ptr_atomic.h.

References std::atomic_compare_exchange_strong_explicit().

Referenced by std::atomic_compare_exchange_weak().

```
3.59.2.10  template<typename _Tp> shared_ptr<_Tp> std::atomic_exchange ( shared_ptr<_Tp> * __p, shared_ptr<_Tp>
    > __r ) [inline]
```

Atomic exchange for shared_ptr objects.

Parameters

\leftarrow __p	A non-null pointer to a shared_ptr object.
\leftarrow __r	New value to store in *__p.

Returns

The original value of *__p

Definition at line 181 of file shared_ptr_atomic.h.

References std::atomic_exchange_explicit().

3.59.2.11 `template<typename _Tp, _Lock_policy _Lp> __shared_ptr<_Tp, _Lp> std::atomic_exchange (__shared_ptr<_Tp, _Lp> * __p, __shared_ptr<_Tp, _Lp> __r) [inline]`

Atomic exchange for `shared_ptr` objects.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__r</code>	New value to store in <code>*__p</code> .

Returns

The original value of `*__p`

Definition at line 200 of file `shared_ptr_atomic.h`.

References `std::atomic_exchange_explicit()`.

3.59.2.12 `template<typename _Tp> shared_ptr<_Tp> std::atomic_exchange_explicit (shared_ptr<_Tp> * __p, shared_ptr<_Tp> __r, memory_order) [inline]`

Atomic exchange for `shared_ptr` objects.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__r</code>	New value to store in <code>*__p</code> .

Returns

The original value of `*__p`

Definition at line 171 of file `shared_ptr_atomic.h`.

3.59.2.13 `template<typename _Tp, _Lock_policy _Lp> __shared_ptr<_Tp, _Lp> std::atomic_exchange_explicit (__shared_ptr<_Tp, _Lp> * __p, __shared_ptr<_Tp, _Lp> __r, memory_order) [inline]`

Atomic exchange for `shared_ptr` objects.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__r</code>	New value to store in <code>*__p</code> .

Returns

The original value of `*__p`

Definition at line 189 of file `shared_ptr_atomic.h`.

Referenced by `std::atomic_exchange()`.

3.59.2.14 `template<typename _Tp, _Lock_policy _Lp> bool std::atomic_is_lock_free (const __shared_ptr<_Tp, _Lp> * __p) [inline]`

Report whether `shared_ptr` atomic operations are lock-free.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
------------------	---

Returns

True if atomic access to `*__p` is lock-free, false otherwise.

Definition at line 71 of file `shared_ptr_atomic.h`.

3.59.2.15 `template<typename _Tp> bool std::atomic_is_lock_free (const shared_ptr<_Tp> * __p) [inline]`

Report whether `shared_ptr` atomic operations are lock-free.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
------------------	---

Returns

True if atomic access to `*__p` is lock-free, false otherwise.

Definition at line 82 of file `shared_ptr_atomic.h`.

3.59.2.16 `template<typename _Tp> shared_ptr<_Tp> std::atomic_load (const shared_ptr<_Tp> * __p) [inline]`

Atomic load for `shared_ptr` objects.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
------------------	---

Returns

*__p

The memory order shall not be `memory_order_release` or `memory_order_acq_rel`.

Definition at line 106 of file `shared_ptr_atomic.h`.

References `std::atomic_load_explicit()`.

3.59.2.17 `template<typename _Tp, _Lock_policy _Lp> __shared_ptr<_Tp, _Lp> std::atomic_load (const __shared_ptr<_Tp, _Lp> * __p) [inline]`

Atomic load for `shared_ptr` objects.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
------------------	---

Returns

*__p

The memory order shall not be `memory_order_release` or `memory_order_acq_rel`.

Definition at line 119 of file `shared_ptr_atomic.h`.

References `std::atomic_load_explicit()`.

3.59.2.18 `template<typename _Tp> shared_ptr<_Tp> std::atomic_load_explicit (const shared_ptr<_Tp> * __p, memory_order) [inline]`

Atomic load for `shared_ptr` objects.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
------------------	---

Returns

*__p

The memory order shall not be `memory_order_release` or `memory_order_acq_rel`.

Definition at line 98 of file `shared_ptr_atomic.h`.

3.59.2.19 `template<typename _Tp, _Lock_policy _Lp> __shared_ptr<_Tp, _Lp> std::atomic_load_explicit (const __shared_ptr<_Tp, _Lp> * __p, memory_order) [inline]`

Atomic load for `shared_ptr` objects.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
------------------	---

Returns

*__p

The memory order shall not be `memory_order_release` or `memory_order_acq_rel`.

Definition at line 111 of file `shared_ptr_atomic.h`.

Referenced by `std::atomic_load()`.

3.59.2.20 `template<typename _Tp> void std::atomic_store (shared_ptr<_Tp> * __p, shared_ptr<_Tp> __r) [inline]`

Atomic store for `shared_ptr` objects.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__r</code>	The value to store.

The memory order shall not be `memory_order_acquire` or `memory_order_acq_rel`.

Definition at line 143 of file `shared_ptr_atomic.h`.

References `std::atomic_store_explicit()`.

3.59.2.21 `template<typename _Tp, _Lock_policy _Lp> void std::atomic_store (__shared_ptr<_Tp, _Lp> * __p, __shared_ptr<_Tp, _Lp> __r) [inline]`

Atomic store for `shared_ptr` objects.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__r</code>	The value to store.

The memory order shall not be `memory_order_acquire` or `memory_order_acq_rel`.

Definition at line 158 of file `shared_ptr_atomic.h`.

References `std::atomic_store_explicit()`.

3.59.2.22 `template<typename _Tp> void std::atomic_store_explicit (shared_ptr<_Tp> * __p, shared_ptr<_Tp> __r, memory_order) [inline]`

Atomic store for `shared_ptr` objects.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__r</code>	The value to store.

The memory order shall not be `memory_order_acquire` or `memory_order_acq_rel`.

Definition at line 134 of file `shared_ptr_atomic.h`.

3.59.2.23 `template<typename _Tp, _Lock_policy _Lp> void std::atomic_store_explicit (__shared_ptr<_Tp, _Lp> * __p, __shared_ptr<_Tp, _Lp> __r, memory_order) [inline]`

Atomic store for `shared_ptr` objects.

Parameters

<code>__p</code>	A non-null pointer to a <code>shared_ptr</code> object.
<code>__r</code>	The value to store.

The memory order shall not be `memory_order_acquire` or `memory_order_acq_rel`.

Definition at line 148 of file `shared_ptr_atomic.h`.

Referenced by `std::atomic_store()`.

3.59.2.24 `template<typename _Del, typename _Tp, _Lock_policy _Lp> _Del* std::get_deleter (const __shared_ptr<_Tp, _Lp> & __p) [inline], [noexcept]`

20.7.2.2.10 `shared_ptr` `get_deleter`

Definition at line 76 of file `bits/shared_ptr.h`.

Referenced by `std::unique_ptr<_Result<_Res>>::operator=()`, `std::unique_ptr<_Tp[], _Dp>::operator=()`, `std::unique_ptr<_Result<_Res>>::reset()`, `std::unique_ptr<_Tp[], _Dp>::reset()`, `std::unique_ptr<_Result<_Res>>::~~unique_ptr()`, and `std::unique_ptr<_Tp[], _Dp>::~~unique_ptr()`.

3.59.2.25 `template<typename _Tp, typename... _Args> shared_ptr<_Tp> std::make_shared (_Args &&... __args)`
`[inline]`

Create an object that is owned by a `shared_ptr`.

Parameters

<code>__args</code>	Arguments for the <code>_Tp</code> object's constructor.
---------------------	--

Returns

A `shared_ptr` that owns the newly created object.

Exceptions

<code>std::bad_alloc</code> , or	an exception thrown from the constructor of <code>_Tp</code> .
----------------------------------	--

Definition at line 632 of file `bits/shared_ptr.h`.

3.59.2.26 `template<typename _Tp, typename... _Args> _MakeUniq<_Tp>::__single_object std::make_unique (_Args &&... __args)` `[inline]`

`std::make_unique` for single objects

Definition at line 790 of file `unique_ptr.h`.

Referenced by `std::make_unique()`.

3.59.2.27 `template<typename _Tp> _MakeUniq<_Tp>::__array std::make_unique (size_t __num)` `[inline]`

`std::make_unique` for arrays of unknown bound

Definition at line 796 of file `unique_ptr.h`.

References `std::make_unique()`.

3.59.2.28 `template<typename _Tp, typename... _Args> _MakeUniq<_Tp>::__invalid_type std::make_unique (_Args && ...)`
`[inline], [delete]`

Disable `std::make_unique` for arrays of known bound.

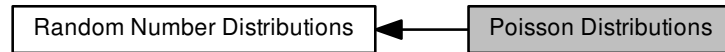
3.59.2.29 `template<typename _Ch, typename _Tr, typename _Tp, _Lock_policy _Lp> std::basic_ostream<_Ch, _Tr>&`
`std::operator<< (std::basic_ostream<_Ch, _Tr> & __os, const __shared_ptr<_Tp, _Lp> & __p)`
`[inline]`

20.7.2.2.11 `shared_ptr` I/O

Definition at line 66 of file `bits/shared_ptr.h`.

3.60 Poisson Distributions

Collaboration diagram for Poisson Distributions:



Classes

- class [std::discrete_distribution< _IntType >](#)
- class [std::exponential_distribution< _RealType >](#)
- class [std::extreme_value_distribution< _RealType >](#)
- class [std::piecewise_constant_distribution< _RealType >](#)
- class [std::piecewise_linear_distribution< _RealType >](#)
- class [std::poisson_distribution< _IntType >](#)
- class [std::weibull_distribution< _RealType >](#)

Functions

- [template<typename _IntType >](#)
[bool std::operator!= \(const std::poisson_distribution< _IntType > &__d1, const std::poisson_distribution< _IntType > &__d2\)](#)
- [template<typename _RealType >](#)
[bool std::operator!= \(const std::exponential_distribution< _RealType > &__d1, const std::exponential_distribution< _RealType > &__d2\)](#)
- [template<typename _RealType >](#)
[bool std::operator!= \(const std::weibull_distribution< _RealType > &__d1, const std::weibull_distribution< _RealType > &__d2\)](#)
- [template<typename _RealType >](#)
[bool std::operator!= \(const std::extreme_value_distribution< _RealType > &__d1, const std::extreme_value_distribution< _RealType > &__d2\)](#)
- [template<typename _IntType >](#)
[bool std::operator!= \(const std::discrete_distribution< _IntType > &__d1, const std::discrete_distribution< _IntType > &__d2\)](#)
- [template<typename _RealType >](#)
[bool std::operator!= \(const std::piecewise_constant_distribution< _RealType > &__d1, const std::piecewise_constant_distribution< _RealType > &__d2\)](#)
- [template<typename _RealType >](#)
[bool std::operator!= \(const std::piecewise_linear_distribution< _RealType > &__d1, const std::piecewise_linear_distribution< _RealType > &__d2\)](#)
- [template<typename _RealType, typename _CharT, typename _Traits >](#)
[std::basic_ostream< _CharT, _Traits > & std::operator<< \(std::basic_ostream< _CharT, _Traits > &__os, const std::exponential_distribution< _RealType > &__x\)](#)

- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const
std::weibull_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const
std::extreme_value_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is, std::←
::exponential_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is, std::←
::weibull_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is, std::←
::extreme_value_distribution< _RealType > &__x)`

3.60.1 Detailed Description

3.60.2 Function Documentation

3.60.2.1 `template<typename _IntType> bool std::operator!=(const std::poisson_distribution< _IntType > &__d1, const
std::poisson_distribution< _IntType > &__d2) [inline]`

Return true if two Poisson distributions are different.

Definition at line 4460 of file random.h.

3.60.2.2 `template<typename _RealType> bool std::operator!=(const std::exponential_distribution< _RealType > &__d1,
const std::exponential_distribution< _RealType > &__d2) [inline]`

Return true if two exponential distributions have different parameters.

Definition at line 4638 of file random.h.

References `std::__detail::operator>>()`.

3.60.2.3 `template<typename _RealType> bool std::operator!=(const std::weibull_distribution< _RealType > &__d1, const
std::weibull_distribution< _RealType > &__d2) [inline]`

Return true if two Weibull distributions have different parameters.

Definition at line 4841 of file random.h.

References `std::__detail::operator>>()`.

3.60.2.4 `template<typename _RealType> bool std::operator!=(const std::extreme_value_distribution< _RealType > &
__d1, const std::extreme_value_distribution< _RealType > &__d2) [inline]`

Return true if two extreme value distributions have different parameters.

Definition at line 5044 of file random.h.

References `std::__detail::operator>>()`.

3.60.2.5 `template<typename _IntType> bool std::operator!= (const std::discrete_distribution< _IntType> &__d1, const std::discrete_distribution< _IntType> &__d2) [inline]`

Return true if two discrete distributions have different parameters.

Definition at line 5304 of file random.h.

3.60.2.6 `template<typename _RealType> bool std::operator!= (const std::piecewise_constant_distribution< _RealType> &__d1, const std::piecewise_constant_distribution< _RealType> &__d2) [inline]`

Return true if two piecewise constant distributions have different parameters.

Definition at line 5571 of file random.h.

3.60.2.7 `template<typename _RealType> bool std::operator!= (const std::piecewise_linear_distribution< _RealType> &__d1, const std::piecewise_linear_distribution< _RealType> &__d2) [inline]`

Return true if two piecewise linear distributions have different parameters.

Definition at line 5841 of file random.h.

3.60.2.8 `template<typename _RealType, typename _CharT, typename _Traits> std::basic_ostream< _CharT, _Traits> & std::operator<< (std::basic_ostream< _CharT, _Traits> &__os, const std::exponential_distribution< _RealType> &__x)`

Inserts a exponential_distribution random number distribution __x into the output stream __os.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A exponential_distribution random number distribution.

Returns

The output stream with the state of __x inserted or in an error state.

Definition at line 1731 of file bits/random.tcc.

References `std::ios_base::flags()`, `std::left()`, and `std::scientific()`.

3.60.2.9 `template<typename _RealType, typename _CharT, typename _Traits> std::basic_ostream< _CharT, _Traits> & std::operator<< (std::basic_ostream< _CharT, _Traits> &__os, const std::weibull_distribution< _RealType> &__x)`

Inserts a weibull_distribution random number distribution __x into the output stream __os.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A weibull_distribution random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

Definition at line 2522 of file bits/random.tcc.

References `std::ios_base::flags()`, `std::left()`, and `std::scientific()`.

```
3.60.2.10 template<typename _RealType, typename _CharT, typename _Traits> std::basic_ostream<_CharT, _Traits> &
std::operator<<( std::basic_ostream<_CharT, _Traits> & __os, const std::extreme_value_distribution<
_RealType> & __x )
```

Inserts a `extreme_value_distribution` random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>extreme_value_distribution</code> random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

Definition at line 2598 of file bits/random.tcc.

References `std::ios_base::flags()`, `std::left()`, and `std::scientific()`.

```
3.60.2.11 template<typename _RealType, typename _CharT, typename _Traits> std::basic_istream<_CharT, _Traits> &
std::operator>>( std::basic_istream<_CharT, _Traits> & __is, std::exponential_distribution<_RealType>
& __x )
```

Extracts a `exponential_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>exponential_distribution</code> random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

Definition at line 1754 of file `bits/random.tcc`.

References `std::dec()`, `std::ios_base::flags()`, `std::normal_distribution<_RealType>::operator()()`, `std::exponential_distribution<_RealType>::param()`, and `std::skipws()`.

3.60.2.12 `template<typename _RealType, typename _CharT, typename _Traits> std::basic_istream<_CharT, _Traits> & std::operator>> (std::basic_istream<_CharT, _Traits> & __is, std::weibull_distribution<_RealType> & __x)`

Extracts a `weibull_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>weibull_distribution</code> random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

Definition at line 2546 of file `bits/random.tcc`.

References `std::dec()`, `std::ios_base::flags()`, `std::log()`, `std::extreme_value_distribution<_RealType>::operator()()`, `std::weibull_distribution<_RealType>::param()`, and `std::skipws()`.

3.60.2.13 `template<typename _RealType, typename _CharT, typename _Traits> std::basic_istream<_CharT, _Traits> & std::operator>> (std::basic_istream<_CharT, _Traits> & __is, std::extreme_value_distribution<_RealType> & __x)`

Extracts a `extreme_value_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>extreme_value_distribution</code> random number generator engine.

Returns

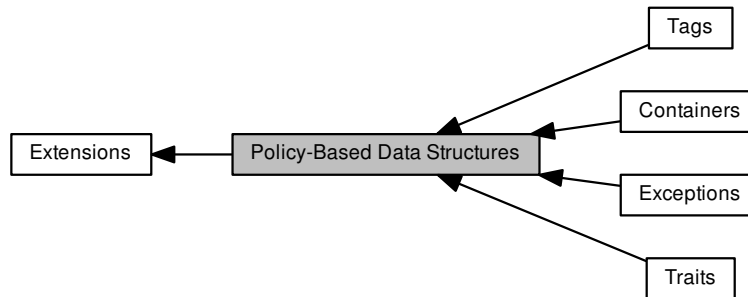
The input stream with `__x` extracted or in an error state.

Definition at line 2622 of file `bits/random.tcc`.

References `std::accumulate()`, `std::back_inserter()`, `std::vector< _Tp, _Alloc >::begin()`, `std::dec()`, `std::piecewise_constant_distribution< _RealType >::densities()`, `std::vector< _Tp, _Alloc >::end()`, `std::basic_ios< _CharT, _Traits >::fill()`, `std::ios_base::flags()`, `std::piecewise_constant_distribution< _RealType >::intervals()`, `std::left()`, `std::lower_bound()`, `std::max()`, `std::discrete_distribution< _IntType >::operator()()`, `std::piecewise_constant_distribution< _RealType >::operator()()`, `std::piecewise_linear_distribution< _RealType >::operator()()`, `std::__detail::operator>>()`, `std::extreme_value_distribution< _RealType >::param()`, `std::discrete_distribution< _IntType >::param()`, `std::piecewise_constant_distribution< _RealType >::param()`, `std::piecewise_linear_distribution< _RealType >::param()`, `std::partial_sum()`, `std::ios_base::precision()`, `std::vector< _Tp, _Alloc >::push_back()`, `std::vector< _Tp, _Alloc >::reserve()`, `std::scientific()`, `std::seed_seq::seed_seq()`, `std::vector< _Tp, _Alloc >::size()`, `std::skipws()`, `std::sqrt()`, and `std::basic_ios< _CharT, _Traits >::widen()`.

3.61 Policy-Based Data Structures

Collaboration diagram for Policy-Based Data Structures:



Modules

- [Containers](#)
- [Exceptions](#)
- [Tags](#)
- [Traits](#)

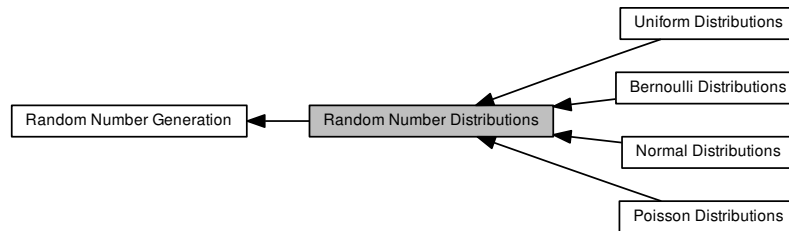
3.61.1 Detailed Description

This is a library of policy-based elementary data structures: associative containers and priority queues. It is designed for high-performance, flexibility, semantic safety, and conformance to the corresponding containers in std (except for some points where it differs by design).

For details, see: http://gcc.gnu.org/onlinedocs/libstdc++/ext/pb_ds/index.html

3.62 Random Number Distributions

Collaboration diagram for Random Number Distributions:



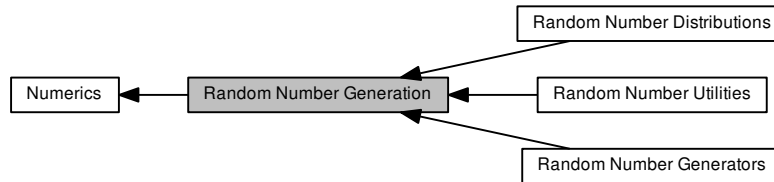
Modules

- [Bernoulli Distributions](#)
- [Normal Distributions](#)
- [Poisson Distributions](#)
- [Uniform Distributions](#)

3.62.1 Detailed Description

3.63 Random Number Generation

Collaboration diagram for Random Number Generation:



Modules

- [Random Number Distributions](#)
- [Random Number Generators](#)
- [Random Number Utilities](#)

Namespaces

- [std::__detail](#)

Functions

- `template<typename _RealType, size_t __bits, typename _UniformRandomNumberGenerator > _RealType std::generate_canonical (_UniformRandomNumberGenerator &__g)`

3.63.1 Detailed Description

A facility for generating random numbers on selected distributions.

3.63.2 Function Documentation

3.63.2.1 `template<typename _RealType, size_t __bits, typename _UniformRandomNumberGenerator > _RealType std::generate_canonical (_UniformRandomNumberGenerator &__g)`

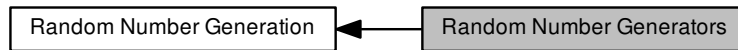
A function template for converting the output of a (integral) uniform random number generator to a floating point result in the range [0-1).

Definition at line 3312 of file `bits/random.tcc`.

References `std::log()`, and `std::min()`.

3.64 Random Number Generators

Collaboration diagram for Random Number Generators:



Classes

- class `std::discard_block_engine< _RandomNumberEngine, __p, __r >`
- class `std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >`
- class `std::linear_congruential_engine< _UIntType, __a, __c, __m >`
- class `std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >`
- class `std::random_device`
- class `std::shuffle_order_engine< _RandomNumberEngine, __k >`
- class `std::subtract_with_carry_engine< _UIntType, __w, __s, __r >`

Typedefs

- typedef `minstd_rand0` **`std::default_random_engine`**
- typedef `shuffle_order_engine< minstd_rand0, 256 >` **`std::knuth_b`**
- typedef `linear_congruential_engine< uint_fast32_t, 48271UL, 0UL, 2147483647UL >` **`std::minstd_rand`**
- typedef `linear_congruential_engine< uint_fast32_t, 16807UL, 0UL, 2147483647UL >` **`std::minstd_rand0`**
- typedef `mersenne_twister_engine< uint_fast32_t, 32, 624, 397, 31, 0x9908b0dfUL, 11, 0xffffffffUL, 7, 0x9d2c5680UL, 15, 0xefc60000UL, 18, 1812433253UL >` **`std::mt19937`**
- typedef `mersenne_twister_engine< uint_fast64_t, 64, 312, 156, 31, 0xb5026f5aa96619e9ULL, 29, 0x5555555555555555ULL, 17, 0x71d67ffeda60000ULL, 37, 0xffff7eee00000000ULL, 43, 6364136223846793005ULL >` **`std::mt19937_64`**
- typedef `discard_block_engine< ranlux24_base, 223, 23 >` **`std::ranlux24`**
- typedef `subtract_with_carry_engine< uint_fast32_t, 24, 10, 24 >` **`std::ranlux24_base`**
- typedef `discard_block_engine< ranlux48_base, 389, 11 >` **`std::ranlux48`**
- typedef `subtract_with_carry_engine< uint_fast64_t, 48, 5, 12 >` **`std::ranlux48_base`**

Functions

- template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m>
bool **`std::operator!=`** (const **`std::linear_congruential_engine< _UIntType, __a, __c, __m >`** &__lhs, const **`std::linear_congruential_engine< _UIntType, __a, __c, __m >`** &__rhs)

- `template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f>`
`bool std::operator!= (const std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f > &__lhs, const std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f > &__rhs)`
- `template<typename _UIntType, size_t __w, size_t __s, size_t __r>`
`bool std::operator!= (const std::subtract_with_carry_engine< _UIntType, __w, __s, __r > &__lhs, const std::subtract_with_carry_engine< _UIntType, __w, __s, __r > &__rhs)`
- `template<typename _RandomNumberEngine, size_t __p, size_t __r>`
`bool std::operator!= (const std::discard_block_engine< _RandomNumberEngine, __p, __r > &__lhs, const std::discard_block_engine< _RandomNumberEngine, __p, __r > &__rhs)`
- `template<typename _RandomNumberEngine, size_t __w, typename _UIntType >`
`bool std::operator!= (const std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > &__lhs, const std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > &__rhs)`
- `template<typename _RandomNumberEngine, size_t __k>`
`bool std::operator!= (const std::shuffle_order_engine< _RandomNumberEngine, __k > &__lhs, const std::shuffle_order_engine< _RandomNumberEngine, __k > &__rhs)`
- `template<typename _RandomNumberEngine, size_t __w, typename _UIntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > &__x)`

3.64.1 Detailed Description

These classes define objects which provide random or pseudorandom numbers, either from a discrete or a continuous interval. The random number generator supplied as a part of this library are all uniform random number generators which provide a sequence of random number uniformly distributed over their range.

A number generator is a function object with an `operator()` that takes zero arguments and returns a number.

A compliant random number generator must satisfy the following requirements.

Table 227 Random Number Generator Requirements

To be documented.

3.64.2 Typedef Documentation

3.64.2.1 `typedef linear_congruential_engine<uint_fast32_t, 48271UL, 0UL, 2147483647UL> std::minstd_rand`

An alternative LCR (Lehmer Generator function).

Definition at line 1519 of file `random.h`.

3.64.2.2 `typedef linear_congruential_engine<uint_fast32_t, 16807UL, 0UL, 2147483647UL> std::minstd_rand0`

The classic Minimum Standard `rand0` of Lewis, Goodman, and Miller.

Definition at line 1513 of file `random.h`.

```
3.64.2.3 typedef mersenne_twister_engine< uint_fast32_t, 32, 624, 397, 31, 0x9908b0dfUL, 11, 0xffffffffUL, 7, 0x9d2c5680UL, 15,
0xefc60000UL, 18, 1812433253UL> std::mt19937
```

The classic Mersenne Twister.

Reference: M. Matsumoto and T. Nishimura, Mersenne Twister: A 623-Dimensionally Equidistributed Uniform Pseudo-Random Number Generator, ACM Transactions on Modeling and Computer Simulation, Vol. 8, No. 1, January 1998, pp 3-30.

Definition at line 1535 of file random.h.

```
3.64.2.4 typedef mersenne_twister_engine< uint_fast64_t, 64, 312, 156, 31, 0xb5026f5aa96619e9ULL, 29, 0x5555555555555555ULL,
17, 0x71d67ffeda60000ULL, 37, 0xfff7eee000000000ULL, 43, 6364136223846793005ULL> std::mt19937_64
```

An alternative Mersenne Twister.

Definition at line 1547 of file random.h.

3.64.3 Function Documentation

```
3.64.3.1 template<typename UIntType, UIntType __a, UIntType __c, UIntType __m> bool std::operator!=
( const std::linear_congruential_engine< UIntType, __a, __c, __m > & __lhs, const
std::linear_congruential_engine< UIntType, __a, __c, __m > & __rhs ) [inline]
```

Compares two linear congruential random number generator objects of the same type for inequality.

Parameters

<code>__lhs</code>	A linear congruential random number generator object.
<code>__rhs</code>	Another linear congruential random number generator object.

Returns

true if the infinite sequences of generated values would be different, false otherwise.

Definition at line 404 of file random.h.

```
3.64.3.2 template<typename UIntType, size_t __w, size_t __n, size_t __m, size_t __r, UIntType __a, size_t __u, UIntType
__d, size_t __s, UIntType __b, size_t __t, UIntType __c, size_t __l, UIntType __f> bool std::operator!= ( const
std::mersenne_twister_engine< UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f > & __lhs,
const std::mersenne_twister_engine< UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f > &
__rhs ) [inline]
```

Compares two % mersenne_twister_engine random number generator objects of the same type for inequality.

Parameters

<code>__lhs</code>	A % mersenne_twister_engine random number generator object.
<code>__rhs</code>	Another % mersenne_twister_engine random number generator object.

Returns

true if the infinite sequences of generated values would be different, false otherwise.

Definition at line 636 of file random.h.

```
3.64.3.3 template<typename UIntType, size_t __w, size_t __s, size_t __r> bool std::operator!= ( const std::subtract_with_carry_engine< UIntType, __w, __s, __r > & __lhs, const std::subtract_with_carry_engine< UIntType, __w, __s, __r > & __rhs ) [inline]
```

Compares two % subtract_with_carry_engine random number generator objects of the same type for inequality.

Parameters

<code>__lhs</code>	A % subtract_with_carry_engine random number generator object.
<code>__rhs</code>	Another % subtract_with_carry_engine random number generator object.

Returns

true if the infinite sequences of generated values would be different, false otherwise.

Definition at line 832 of file random.h.

```
3.64.3.4 template<typename RandomNumberEngine, size_t __p, size_t __r> bool std::operator!= ( const std::discard_block_engine< RandomNumberEngine, __p, __r > & __lhs, const std::discard_block_engine< RandomNumberEngine, __p, __r > & __rhs ) [inline]
```

Compares two discard_block_engine random number generator objects of the same type for inequality.

Parameters

<code>__lhs</code>	A discard_block_engine random number generator object.
<code>__rhs</code>	Another discard_block_engine random number generator object.

Returns

true if the infinite sequences of generated values would be different, false otherwise.

Definition at line 1054 of file random.h.

```
3.64.3.5 template<typename _RandomNumberEngine , size_t __w, typename _UIntType > bool std::operator!=
( const std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > & __lhs, const
  std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > & __rhs ) [inline]
```

Compares two `independent_bits_engine` random number generator objects of the same type for inequality.

Parameters

<code>__lhs</code>	A <code>independent_bits_engine</code> random number generator object.
<code>__rhs</code>	Another <code>independent_bits_engine</code> random number generator object.

Returns

true if the infinite sequences of generated values would be different, false otherwise.

Definition at line 1250 of file `random.h`.

```
3.64.3.6 template<typename _RandomNumberEngine , size_t __k> bool std::operator!= ( const std::shuffle_order_engine<
  _RandomNumberEngine, __k > & __lhs, const std::shuffle_order_engine< _RandomNumberEngine, __k > & __rhs )
[inline]
```

Compares two `shuffle_order_engine` random number generator objects of the same type for inequality.

Parameters

<code>__lhs</code>	A <code>shuffle_order_engine</code> random number generator object.
<code>__rhs</code>	Another <code>shuffle_order_engine</code> random number generator object.

Returns

true if the infinite sequences of generated values would be different, false otherwise.

Definition at line 1502 of file `random.h`.

```
3.64.3.7 template<typename _RandomNumberEngine , size_t __w, typename _UIntType , typename _CharT , typename _Traits >
  std::basic_ostream< _CharT, _Traits>& std::operator<< ( std::basic_ostream< _CharT, _Traits > & __os, const
  std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > & __x )
```

Inserts the current state of a `independent_bits_engine` random number generator engine `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>independent_bits_engine</code> random number generator engine.

Returns

The output stream with the state of `__x` inserted or in an error state.

Definition at line 1269 of file `random.h`.

References `std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType >::base()`.

3.65 Random Number Utilities

Collaboration diagram for Random Number Utilities:



Classes

- class [std::seed_seq](#)

3.65.1 Detailed Description

3.66 Rational Arithmetic

Collaboration diagram for Rational Arithmetic:



Classes

- struct `std::ratio< _Num, _Den >`
- struct `std::ratio_equal< _R1, _R2 >`
- struct `std::ratio_not_equal< _R1, _R2 >`

Typedefs

- template<typename _R1, typename _R2 >
using `std::ratio_divide` = typename `__ratio_divide< _R1, _R2 >::type`
- template<typename _R1, typename _R2 >
using `std::ratio_multiply` = typename `__ratio_multiply< _R1, _R2 >::type`
- typedef `ratio< num, den > std::ratio< _Num, _Den >::type`
- typedef `ratio< __safe_multiply<(_R1::num/__gcd1),(_R2::num/__gcd2)>::value, __safe_multiply<(_R1::den/__gcd2),(_R2::den/__gcd1)>::value > std::ratio_multiply< _R1, _R2 >::type`
- typedef `__ratio_multiply< _R1, ratio< _R2::den, _R2::num > >::type std::ratio_divide< _R1, _R2 >::type`

Variables

- static constexpr uintmax_t `std::big_add< __hi1, __lo1, __hi2, __lo2 >::hi`
- static constexpr uintmax_t `std::big_sub< __hi1, __lo1, __hi2, __lo2 >::hi`
- static constexpr uintmax_t `std::big_mul< __x, __y >::hi`
- static constexpr uintmax_t `std::big_add< __hi1, __lo1, __hi2, __lo2 >::lo`
- static constexpr uintmax_t `std::big_sub< __hi1, __lo1, __hi2, __lo2 >::lo`
- static constexpr uintmax_t `std::big_mul< __x, __y >::lo`
- static constexpr uintmax_t `std::big_div_impl< __n1, __n0, __d >::quot`
- static constexpr uintmax_t `std::big_div< __n1, __n0, __d >::quot_hi`
- static constexpr uintmax_t `std::big_div< __n1, __n0, __d >::quot_lo`
- static constexpr uintmax_t `std::big_div_impl< __n1, __n0, __d >::rem`
- static constexpr uintmax_t `std::big_div< __n1, __n0, __d >::rem`
- static constexpr intmax_t `std::ratio< _Num, _Den >::den`
- static constexpr intmax_t `std::ratio_multiply< _R1, _R2 >::den`
- static constexpr intmax_t `std::ratio_divide< _R1, _R2 >::den`
- static constexpr intmax_t `std::ratio< _Num, _Den >::num`
- static constexpr intmax_t `std::ratio_multiply< _R1, _R2 >::num`
- static constexpr intmax_t `std::ratio_divide< _R1, _R2 >::num`
- static const intmax_t `std::safe_multiply< _Pn, _Qn >::value`

3.66.1 Detailed Description

Compile time representation of finite rational numbers.

3.66.2 Typedef Documentation

3.66.2.1 `template<typename _R1, typename _R2 > using std::ratio_divide = typedef typename __ratio_divide<_R1, _R2>::type`

ratio_divide

Definition at line 336 of file ratio.

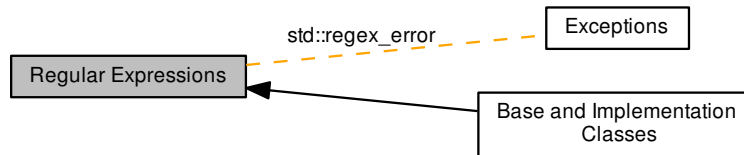
3.66.2.2 `template<typename _R1, typename _R2 > using std::ratio_multiply = typedef typename __ratio_multiply<_R1, _R2>::type`

ratio_multiply

Definition at line 313 of file ratio.

3.67 Regular Expressions

Collaboration diagram for Regular Expressions:



Modules

- [Base and Implementation Classes](#)

Namespaces

- [std::regex_constants](#)

Classes

- class [std::basic_regex< _Ch_type, _Rx_traits >](#)
- class [std::match_results< _Bi_iter, _Alloc >](#)
- class [std::regex_error](#)
- class [std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >](#)
- class [std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >](#)
- struct [std::regex_traits< _Ch_type >](#)
- class [std::sub_match< _Biter >](#)

Typedefs

- template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >
using **std::__sub_match_string** = basic_string< typename iterator_traits< _Bi_iter >::value_type, _Ch_traits, _Ch_alloc >
- typedef match_results< const char * > **std::cmatch**
- typedef regex_iterator< const char * > **std::cregex_iterator**
- typedef regex_token_iterator< const char * > [std::cregex_token_iterator](#)
- typedef sub_match< const char * > [std::csub_match](#)
- typedef basic_regex< char > [std::regex](#)
- typedef match_results< string::const_iterator > **std::smatch**
- typedef regex_iterator< string::const_iterator > **std::sregex_iterator**
- typedef regex_token_iterator< string::const_iterator > [std::sregex_token_iterator](#)

- typedef sub_match< string::const_iterator > [std::ssub_match](#)
- typedef match_results< const wchar_t * > [std::wcmatch](#)
- typedef regex_iterator< const wchar_t * > [std::wcregex_iterator](#)
- typedef regex_token_iterator< const wchar_t * > [std::wcregex_token_iterator](#)
- typedef sub_match< const wchar_t * > [std::wcsub_match](#)
- typedef basic_regex< wchar_t > [std::wregex](#)
- typedef match_results< wstring::const_iterator > [std::wsmatch](#)
- typedef regex_iterator< wstring::const_iterator > [std::wsregex_iterator](#)
- typedef regex_token_iterator< wstring::const_iterator > [std::wsregex_token_iterator](#)
- typedef sub_match< wstring::const_iterator > [std::wssub_match](#)

Functions

- template<typename _Bilter >
bool [std::operator!=](#) (const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)
- template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >
bool [std::operator!=](#) (const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__lhs, const sub_match< _Bi_iter > &__rhs)
- template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >
bool [std::operator!=](#) (const sub_match< _Bi_iter > &__lhs, const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__rhs)
- template<typename _Bi_iter >
bool [std::operator!=](#) (typename iterator_traits< _Bi_iter >::value_type const *__lhs, const sub_match< _Bi_iter > &__rhs)
- template<typename _Bi_iter >
bool [std::operator!=](#) (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const *__rhs)
- template<typename _Bi_iter >
bool [std::operator!=](#) (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)
- template<typename _Bi_iter >
bool [std::operator!=](#) (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)
- template<typename _Bi_iter, class _Alloc >
bool [std::operator!=](#) (const match_results< _Bi_iter, _Alloc > &__m1, const match_results< _Bi_iter, _Alloc > &__m2)
- template<typename _Bilter >
bool [std::operator<](#) (const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)
- template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >
bool [std::operator<](#) (const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__lhs, const sub_match< _Bi_iter > &__rhs)
- template<typename _Bi_iter, class _Ch_traits, class _Ch_alloc >
bool [std::operator<](#) (const sub_match< _Bi_iter > &__lhs, const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__rhs)
- template<typename _Bi_iter >
bool [std::operator<](#) (typename iterator_traits< _Bi_iter >::value_type const *__lhs, const sub_match< _Bi_iter > &__rhs)
- template<typename _Bi_iter >
bool [std::operator<](#) (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const *__rhs)

- `template<typename _Bi_iter >`
`bool std::operator< (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator< (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Ch_type, typename _Ch_traits, typename _Bi_iter >`
`basic_ostream< _Ch_type, _Ch_traits > & std::operator<< (basic_ostream< _Ch_type, _Ch_traits > &__os, const sub_match< _Bi_iter > &__m)`
- `template<typename _Bilter >`
`bool std::operator<= (const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)`
- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
`bool std::operator<= (const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter, class _Ch_traits, class _Ch_alloc >`
`bool std::operator<= (const sub_match< _Bi_iter > &__lhs, const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator<= (typename iterator_traits< _Bi_iter >::value_type const *__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator<= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const *__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator<= (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator<= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Bilter >`
`bool std::operator== (const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)`
- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
`bool std::operator== (const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
`bool std::operator== (const sub_match< _Bi_iter > &__lhs, const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator== (typename iterator_traits< _Bi_iter >::value_type const *__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator== (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const *__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator== (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator== (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Bi_iter, typename _Alloc >`
`bool std::operator== (const match_results< _Bi_iter, _Alloc > &__m1, const match_results< _Bi_iter, _Alloc > &__m2)`

- `template<typename _Bilter >`
`bool std::operator> (const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)`
- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
`bool std::operator> (const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter, class _Ch_traits, class _Ch_alloc >`
`bool std::operator> (const sub_match< _Bi_iter > &__lhs, const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator> (typename iterator_traits< _Bi_iter >::value_type const *__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator> (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const *__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator> (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator> (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Bilter >`
`bool std::operator>= (const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)`
- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
`bool std::operator>= (const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter, class _Ch_traits, class _Ch_alloc >`
`bool std::operator>= (const sub_match< _Bi_iter > &__lhs, const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator>= (typename iterator_traits< _Bi_iter >::value_type const *__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator>= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const *__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator>= (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator>= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Ch_type, typename _Rx_traits >`
`void std::swap (basic_regex< _Ch_type, _Rx_traits > &__lhs, basic_regex< _Ch_type, _Rx_traits > &__rhs)`
- `template<typename _Bi_iter, typename _Alloc >`
`void std::swap (match_results< _Bi_iter, _Alloc > &__lhs, match_results< _Bi_iter, _Alloc > &__rhs)`

Matching, Searching, and Replacing

- `template<typename _Bi_iter, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_match (_Bi_iter __s, _Bi_iter __e, match_results< _Bi_iter, _Alloc > &__m, const basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Bi_iter, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_match (_Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`

- `template<typename _Ch_type, typename _Alloc, typename _Rx_traits >`
`bool std::regex_match (const _Ch_type * __s, match_results< const _Ch_type *, _Alloc > & __m, const basic_regex< _Ch_type, _Rx_traits > & __re, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_match (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > & __s, match_results< typename basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > & __m, const basic_regex< _Ch_type, _Rx_traits > & __re, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_match (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &&, match_results< typename basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &, const basic_regex< _Ch_type, _Rx_traits > &, regex_constants::match_flag_type=regex_constants::match_default)=delete`
- `template<typename _Ch_type, class _Rx_traits >`
`bool std::regex_match (const _Ch_type * __s, const basic_regex< _Ch_type, _Rx_traits > & __re, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Str_allocator, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_match (const basic_string< _Ch_type, _Ch_traits, _Str_allocator > & __s, const basic_regex< _Ch_type, _Rx_traits > & __re, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Bi_iter, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (_Bi_iter __s, _Bi_iter __e, match_results< _Bi_iter, _Alloc > & __m, const basic_regex< _Ch_type, _Rx_traits > & __re, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Bi_iter, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (_Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type, _Rx_traits > & __re, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_type, class _Alloc, class _Rx_traits >`
`bool std::regex_search (const _Ch_type * __s, match_results< const _Ch_type *, _Alloc > & __m, const basic_regex< _Ch_type, _Rx_traits > & __e, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (const _Ch_type * __s, const basic_regex< _Ch_type, _Rx_traits > & __e, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _String_allocator, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (const basic_string< _Ch_type, _Ch_traits, _String_allocator > & __s, const basic_regex< _Ch_type, _Rx_traits > & __e, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > & __s, match_results< typename basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > & __m, const basic_regex< _Ch_type, _Rx_traits > & __e, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &&, match_results< typename basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &, const basic_regex< _Ch_type, _Rx_traits > &, regex_constants::match_flag_type=regex_constants::match_default)=delete`
- `template<typename _Out_iter, typename _Bi_iter, typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa >`
`_Out_iter std::regex_replace (_Out_iter __out, _Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type, _Rx_traits > & __e, const basic_string< _Ch_type, _St, _Sa > & __fmt, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Out_iter, typename _Bi_iter, typename _Rx_traits, typename _Ch_type >`
`_Out_iter std::regex_replace (_Out_iter __out, _Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type, _Rx_traits > & __e, const _Ch_type * __fmt, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa, typename _Fst, typename _Fsa >`
`basic_string< _Ch_type, _St, _Sa > std::regex_replace (const basic_string< _Ch_type, _St, _Sa > & __s, const basic_regex< _Ch_type, _Rx_traits > & __e, const basic_string< _Ch_type, _Fst, _Fsa > & __fmt, regex_constants::match_flag_type __f=regex_constants::match_default)`

- `template<typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa >
basic_string< _Ch_type, _St, _Sa > std::regex_replace (const basic_string< _Ch_type, _St, _Sa > &__s, const
basic_regex< _Ch_type, _Rx_traits > &__e, const _Ch_type *__fmt, regex_constants::match_flag_type __↵
flags=regex_constants::match_default)`
- `template<typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa >
basic_string< _Ch_type > std::regex_replace (const _Ch_type *__s, const basic_regex< _Ch_type, _Rx_traits
> &__e, const basic_string< _Ch_type, _St, _Sa > &__fmt, regex_constants::match_flag_type __flags=regex↵
_constants::match_default)`
- `template<typename _Rx_traits, typename _Ch_type >
basic_string< _Ch_type > std::regex_replace (const _Ch_type *__s, const basic_regex< _Ch_type, _Rx_traits
> &__e, const _Ch_type *__fmt, regex_constants::match_flag_type __flags=regex_constants::match_default)`

3.67.1 Detailed Description

A facility for performing regular expression pattern matching.

3.67.2 Typedef Documentation

3.67.2.1 `typedef regex_token_iterator<const char*> std::cregex_token_iterator`

Token iterator for C-style NULL-terminated strings.

Definition at line 2783 of file `regex.h`.

3.67.2.2 `typedef sub_match<const char*> std::csub_match`

Standard regex submatch over a C-style null-terminated string.

Definition at line 917 of file `regex.h`.

3.67.2.3 `typedef basic_regex<char> std::regex`

Standard regular expressions.

Definition at line 789 of file `regex.h`.

3.67.2.4 `typedef regex_token_iterator<string::const_iterator> std::sregex_token_iterator`

Token iterator for standard strings.

Definition at line 2786 of file `regex.h`.

3.67.2.5 `typedef sub_match<string::const_iterator> std::ssub_match`

Standard regex submatch over a standard string.

Definition at line 920 of file `regex.h`.

3.67.2.6 `typedef regex_token_iterator<const wchar_t*> std::wcregex_token_iterator`

Token iterator for C-style NULL-terminated wide strings.

Definition at line 2790 of file `regex.h`.

3.67.2.7 `typedef sub_match<const wchar_t*> std::wcsub_match`

Regex submatch over a C-style null-terminated wide string.

Definition at line 924 of file `regex.h`.

3.67.2.8 `typedef basic_regex<wchar_t> std::wregex`

Standard wide-character regular expressions.

Definition at line 793 of file `regex.h`.

3.67.2.9 `typedef regex_token_iterator<wstring::const_iterator> std::wsregex_token_iterator`

Token iterator for standard wide-character strings.

Definition at line 2793 of file `regex.h`.

3.67.2.10 `typedef sub_match<wstring::const_iterator> std::wssub_match`

Regex submatch over a standard wide string.

Definition at line 927 of file `regex.h`.

3.67.3 Function Documentation

3.67.3.1 `template<typename _Bilter > bool std::operator!=(const sub_match<_Bilter > &__lhs, const sub_match<_Bilter > &__rhs) [inline]`

Tests the inequivalence of two regular expression submatches.

Parameters

<code>__lhs</code>	First regular expression submatch.
<code>__rhs</code>	Second regular expression submatch.

Returns

true if `__lhs` is not equivalent to `__rhs`, false otherwise.

Definition at line 951 of file regex.h.

References `std::sub_match<_Bilter>::compare()`.

```
3.67.3.2 template<typename _Bi_iter , typename _Ch_traits , typename _Ch_alloc > bool std::operator!= ( const
    __sub_match_string<_Bi_iter, _Ch_traits, _Ch_alloc > & __lhs, const sub_match<_Bi_iter > & __rhs )
    [inline]
```

Tests the inequivalence of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` is not equivalent to `__rhs`, false otherwise.

Definition at line 1029 of file regex.h.

```
3.67.3.3 template<typename _Bi_iter , typename _Ch_traits , typename _Ch_alloc > bool std::operator!= ( const sub_match<
    _Bi_iter > & __lhs, const __sub_match_string<_Bi_iter, _Ch_traits, _Ch_alloc > & __rhs ) [inline]
```

Tests the inequivalence of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A string.

Returns

true if `__lhs` is not equivalent to `__rhs`, false otherwise.

Definition at line 1109 of file regex.h.

```
3.67.3.4 template<typename _Bi_iter > bool std::operator!= ( typename iterator_traits<_Bi_iter >::value_type const * __lhs,
    const sub_match<_Bi_iter > & __rhs ) [inline]
```

Tests the inequivalence of an iterator value and a regular expression submatch.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A string.

Returns

true if `__lhs` is not equivalent to `__rhs`, false otherwise.

Definition at line 1186 of file `regex.h`.

```
3.67.3.5  template<typename _Bi_iter > bool std::operator!=( const sub_match< _Bi_iter > & __lhs, typename iterator_traits<
    _Bi_iter >::value_type const * __rhs ) [inline]
```

Tests the inequivalence of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A pointer to a string.

Returns

true if `__lhs` is not equivalent to `__rhs`, false otherwise.

Definition at line 1260 of file `regex.h`.

```
3.67.3.6  template<typename _Bi_iter > bool std::operator!=( typename iterator_traits< _Bi_iter >::value_type const & __lhs,
    const sub_match< _Bi_iter > & __rhs ) [inline]
```

Tests the inequivalence of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` is not equivalent to `__rhs`, false otherwise.

Definition at line 1337 of file `regex.h`.

```
3.67.3.7  template<typename _Bi_iter > bool std::operator!=( const sub_match< _Bi_iter > & __lhs, typename iterator_traits<
    _Bi_iter >::value_type const & __rhs ) [inline]
```

Tests the inequivalence of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A const string reference.

Returns

true if `__lhs` is not equivalent to `__rhs`, false otherwise.

Definition at line 1417 of file `regex.h`.

```
3.67.3.8  template<typename _Bi_iter, class _Alloc > bool std::operator!=( const match_results< _Bi_iter, _Alloc > & __m1,
        const match_results< _Bi_iter, _Alloc > & __m2 )  [inline]
```

Compares two `match_results` for inequality.

Returns

true if the two objects do not refer to the same match, false otherwise.

Definition at line 1944 of file `regex.h`.

```
3.67.3.9  template<typename _Bilter > bool std::operator< ( const sub_match< _Bilter > & __lhs, const sub_match< _Bilter
        > & __rhs )  [inline]
```

Tests the ordering of two regular expression submatches.

Parameters

<code>__lhs</code>	First regular expression submatch.
<code>__rhs</code>	Second regular expression submatch.

Returns

true if `__lhs` precedes `__rhs`, false otherwise.

Definition at line 962 of file `regex.h`.

References `std::sub_match< _Bilter >::compare()`.

```
3.67.3.10 template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc > bool std::operator< ( const
        __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > & __lhs, const sub_match< _Bi_iter > & __rhs )
        [inline]
```

Tests the ordering of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` precedes `__rhs`, false otherwise.

Definition at line 1041 of file `regex.h`.

References `std::sub_match<_Biter>::compare()`.

3.67.3.11 `template<typename _Bi_iter, class _Ch_traits, class _Ch_alloc> bool std::operator< (const sub_match<_Bi_iter> & __lhs, const __sub_match_string<_Bi_iter, _Ch_traits, _Ch_alloc> & __rhs) [inline]`

Tests the ordering of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A string.

Returns

true if `__lhs` precedes `__rhs`, false otherwise.

Definition at line 1121 of file `regex.h`.

References `std::sub_match<_Biter>::compare()`.

3.67.3.12 `template<typename _Bi_iter> bool std::operator< (typename iterator_traits<_Bi_iter>::value_type const * __lhs, const sub_match<_Bi_iter> & __rhs) [inline]`

Tests the ordering of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` precedes `__rhs`, false otherwise.

Definition at line 1198 of file `regex.h`.

References `std::sub_match<_Biter>::compare()`.

3.67.3.13 `template<typename _Bi_iter> bool std::operator< (const sub_match<_Bi_iter> & __lhs, typename iterator_traits<_Bi_iter>::value_type const * __rhs) [inline]`

Tests the ordering of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A string.

Returns

true if `__lhs` precedes `__rhs`, false otherwise.

Definition at line 1272 of file `regex.h`.

References `std::basic_string<_CharT, _Traits, _Alloc>::compare()`.

3.67.3.14 `template<typename _Bi_iter> bool std::operator< (typename iterator_traits<_Bi_iter>::value_type const & __lhs, const sub_match<_Bi_iter> & __rhs) [inline]`

Tests the ordering of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` precedes `__rhs`, false otherwise.

Definition at line 1349 of file `regex.h`.

References `std::sub_match<_Biter>::compare()`.

3.67.3.15 `template<typename _Bi_iter> bool std::operator< (const sub_match<_Bi_iter> & __lhs, typename iterator_traits<_Bi_iter>::value_type const & __rhs) [inline]`

Tests the ordering of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A const string reference.

Returns

true if `__lhs` precedes `__rhs`, false otherwise.

Definition at line 1429 of file `regex.h`.

References `std::sub_match<_Biter>::compare()`.

3.67.3.16 `template<typename _Ch_type, typename _Ch_traits, typename _Bi_iter > basic_ostream<_Ch_type, _Ch_traits>& std::operator<< (basic_ostream<_Ch_type, _Ch_traits> & __os, const sub_match<_Bi_iter> & __m) [inline]`

Inserts a matched string into an output stream.

Parameters

<code>__os</code>	The output stream.
<code>__m</code>	A submatch string.

Returns

the output stream with the submatch string inserted.

Definition at line 1483 of file regex.h.

References `std::sub_match<_Bilter>::str()`.

3.67.3.17 `template<typename _Bilter > bool std::operator<= (const sub_match<_Bilter> & __lhs, const sub_match<_Bilter> & __rhs) [inline]`

Tests the ordering of two regular expression submatches.

Parameters

<code>__lhs</code>	First regular expression submatch.
<code>__rhs</code>	Second regular expression submatch.

Returns

true if `__lhs` does not succeed `__rhs`, false otherwise.

Definition at line 973 of file regex.h.

References `std::sub_match<_Bilter>::compare()`.

3.67.3.18 `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc > bool std::operator<= (const __sub_match_string<_Bi_iter, _Ch_traits, _Ch_alloc> & __lhs, const sub_match<_Bi_iter> & __rhs) [inline]`

Tests the ordering of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` does not succeed `__rhs`, false otherwise.

Definition at line 1080 of file `regex.h`.

```
3.67.3.19 template<typename _Bi_iter, class _Ch_traits, class _Ch_alloc> bool std::operator<= ( const sub_match< _Bi_iter
> & __lhs, const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc> & __rhs ) [inline]
```

Tests the ordering of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A string.

Returns

true if `__lhs` does not succeed `__rhs`, false otherwise.

Definition at line 1160 of file `regex.h`.

```
3.67.3.20 template<typename _Bi_iter> bool std::operator<= ( typename iterator_traits< _Bi_iter>::value_type const * __lhs,
const sub_match< _Bi_iter> & __rhs ) [inline]
```

Tests the ordering of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` does not succeed `__rhs`, false otherwise.

Definition at line 1234 of file `regex.h`.

```
3.67.3.21 template<typename _Bi_iter> bool std::operator<= ( const sub_match< _Bi_iter> & __lhs, typename
iterator_traits< _Bi_iter>::value_type const * __rhs ) [inline]
```

Tests the ordering of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A string.

Returns

true if `__lhs` does not succeed `__rhs`, false otherwise.

Definition at line 1308 of file `regex.h`.

3.67.3.22 `template<typename _Bi_iter> bool std::operator<= (typename iterator_traits<_Bi_iter>::value_type const & __lhs, const sub_match<_Bi_iter> & __rhs) [inline]`

Tests the ordering of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` does not succeed `__rhs`, false otherwise.

Definition at line 1388 of file `regex.h`.

3.67.3.23 `template<typename _Bi_iter> bool std::operator<= (const sub_match<_Bi_iter> & __lhs, typename iterator_traits<_Bi_iter>::value_type const & __rhs) [inline]`

Tests the ordering of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A const string reference.

Returns

true if `__lhs` does not succeed `__rhs`, false otherwise.

Definition at line 1468 of file `regex.h`.

3.67.3.24 `template<typename _Biter> bool std::operator== (const sub_match<_Biter> & __lhs, const sub_match<_Biter> & __rhs) [inline]`

Tests the equivalence of two regular expression submatches.

Parameters

<code>__lhs</code>	First regular expression submatch.
<code>__rhs</code>	Second regular expression submatch.

Returns

true if `__lhs` is equivalent to `__rhs`, false otherwise.

Definition at line 940 of file `regex.h`.

References `std::sub_match<_Bilter>::compare()`.

```
3.67.3.25 template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc> bool std::operator==( const
    __sub_match_string<_Bi_iter, _Ch_traits, _Ch_alloc> & __lhs, const sub_match<_Bi_iter> & __rhs )
    [inline]
```

Tests the equivalence of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` is equivalent to `__rhs`, false otherwise.

Definition at line 1013 of file `regex.h`.

References `std::sub_match<_Bilter>::compare()`, `std::basic_string<_CharT, _Traits, _Alloc>::data()`, and `std::basic_string<_CharT, _Traits, _Alloc>::size()`.

```
3.67.3.26 template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc> bool std::operator==( const sub_match<
    _Bi_iter> & __lhs, const __sub_match_string<_Bi_iter, _Ch_traits, _Ch_alloc> & __rhs ) [inline]
```

Tests the equivalence of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A string.

Returns

true if `__lhs` is equivalent to `__rhs`, false otherwise.

Definition at line 1093 of file `regex.h`.

References `std::sub_match<_Bilter>::compare()`, `std::basic_string<_CharT, _Traits, _Alloc>::data()`, and `std::basic_string<_CharT, _Traits, _Alloc>::size()`.

3.67.3.27 `template<typename _Bi_iter> bool std::operator==(typename iterator_traits<_Bi_iter>::value_type const * __lhs,
const sub_match<_Bi_iter> & __rhs) [inline]`

Tests the equivalence of a C string and a regular expression submatch.

Parameters

<code>__lhs</code>	A C string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` is equivalent to `__rhs`, false otherwise.

Definition at line 1173 of file `regex.h`.

References `std::sub_match<_Biter>::compare()`.

3.67.3.28 `template<typename _Bi_iter> bool std::operator==(const sub_match<_Bi_iter> & __lhs, typename iterator_traits<_Bi_iter>::value_type const * __rhs) [inline]`

Tests the equivalence of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A pointer to a string?

Returns

true if `__lhs` is equivalent to `__rhs`, false otherwise.

Definition at line 1247 of file `regex.h`.

References `std::sub_match<_Biter>::compare()`.

3.67.3.29 `template<typename _Bi_iter> bool std::operator==(typename iterator_traits<_Bi_iter>::value_type const & __lhs, const sub_match<_Bi_iter> & __rhs) [inline]`

Tests the equivalence of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` is equivalent to `__rhs`, false otherwise.

Definition at line 1321 of file `regex.h`.

References `std::sub_match<_Biter>::compare()`.

3.67.3.30 `template<typename _Bi_iter > bool std::operator==(const sub_match< _Bi_iter > & __lhs, typename iterator_traits< _Bi_iter >::value_type const & __rhs) [inline]`

Tests the equivalence of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A const string reference.

Returns

true if `__lhs` is equivalent to `__rhs`, false otherwise.

Definition at line 1401 of file `regex.h`.

References `std::sub_match< _Biliter >::compare()`.

3.67.3.31 `template<typename _Bi_iter, typename _Alloc > bool std::operator==(const match_results< _Bi_iter, _Alloc > & __m1, const match_results< _Bi_iter, _Alloc > & __m2) [inline]`

Compares two `match_results` for equality.

Returns

true if the two objects refer to the same match, false otherwise.

Definition at line 1920 of file `regex.h`.

References `std::match_results< _Bi_iter, _Alloc >::begin()`, `std::match_results< _Bi_iter, _Alloc >::empty()`, `std::match_results< _Bi_iter, _Alloc >::end()`, `std::equal()`, `std::match_results< _Bi_iter, _Alloc >::prefix()`, `std::match_results< _Bi_iter, _Alloc >::ready()`, `std::match_results< _Bi_iter, _Alloc >::size()`, and `std::match_results< _Bi_iter, _Alloc >::suffix()`.

3.67.3.32 `template<typename _Biliter > bool std::operator> (const sub_match< _Biliter > & __lhs, const sub_match< _Biliter > & __rhs) [inline]`

Tests the ordering of two regular expression submatches.

Parameters

<code>__lhs</code>	First regular expression submatch.
<code>__rhs</code>	Second regular expression submatch.

Returns

true if `__lhs` succeeds `__rhs`, false otherwise.

Definition at line 995 of file regex.h.

References `std::sub_match<_Biter>::compare()`.

```
3.67.3.33 template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc> bool std::operator> ( const
    __sub_match_string<_Bi_iter, _Ch_traits, _Ch_alloc> & __lhs, const sub_match<_Bi_iter> & __rhs )
    [inline]
```

Tests the ordering of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` succeeds `__rhs`, false otherwise.

Definition at line 1056 of file regex.h.

```
3.67.3.34 template<typename _Bi_iter, class _Ch_traits, class _Ch_alloc> bool std::operator> ( const sub_match<_Bi_iter>
    & __lhs, const __sub_match_string<_Bi_iter, _Ch_traits, _Ch_alloc> & __rhs ) [inline]
```

Tests the ordering of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A string.

Returns

true if `__lhs` succeeds `__rhs`, false otherwise.

Definition at line 1136 of file regex.h.

```
3.67.3.35 template<typename _Bi_iter> bool std::operator> ( typename iterator_traits<_Bi_iter>::value_type const * __lhs,
    const sub_match<_Bi_iter> & __rhs ) [inline]
```

Tests the ordering of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` succeeds `__rhs`, false otherwise.

Definition at line 1210 of file `regex.h`.

3.67.3.36 `template<typename _Bi_iter > bool std::operator> (const sub_match<_Bi_iter > & __lhs, typename iterator_traits<_Bi_iter >::value_type const * __rhs) [inline]`

Tests the ordering of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A string.

Returns

true if `__lhs` succeeds `__rhs`, false otherwise.

Definition at line 1284 of file `regex.h`.

3.67.3.37 `template<typename _Bi_iter > bool std::operator> (typename iterator_traits<_Bi_iter >::value_type const & __lhs, const sub_match<_Bi_iter > & __rhs) [inline]`

Tests the ordering of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` succeeds `__rhs`, false otherwise.

Definition at line 1364 of file `regex.h`.

3.67.3.38 `template<typename _Bi_iter > bool std::operator> (const sub_match<_Bi_iter > & __lhs, typename iterator_traits<_Bi_iter >::value_type const & __rhs) [inline]`

Tests the ordering of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A const string reference.

Returns

true if `__lhs` succeeds `__rhs`, false otherwise.

Definition at line 1444 of file `regex.h`.

```
3.67.3.39 template<typename _Bilter > bool std::operator>= ( const sub_match< _Bilter > & __lhs, const sub_match<
    _Bilter > & __rhs ) [inline]
```

Tests the ordering of two regular expression submatches.

Parameters

<code>__lhs</code>	First regular expression submatch.
<code>__rhs</code>	Second regular expression submatch.

Returns

true if `__lhs` does not precede `__rhs`, false otherwise.

Definition at line 984 of file `regex.h`.

References `std::sub_match< _Bilter >::compare()`.

```
3.67.3.40 template<typename _Bi_iter , typename _Ch_traits , typename _Ch_alloc > bool std::operator>= ( const
    __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > & __lhs, const sub_match< _Bi_iter > & __rhs )
    [inline]
```

Tests the ordering of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` does not precede `__rhs`, false otherwise.

Definition at line 1068 of file `regex.h`.

```
3.67.3.41 template<typename _Bi_iter , class _Ch_traits , class _Ch_alloc > bool std::operator>= ( const sub_match< _Bi_iter
    > & __lhs, const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > & __rhs ) [inline]
```

Tests the ordering of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A string.

Returns

true if `__lhs` does not precede `__rhs`, false otherwise.

Definition at line 1148 of file `regex.h`.

```
3.67.3.42 template<typename _Bi_iter> bool std::operator>= ( typename iterator_traits<_Bi_iter>::value_type const * __lhs,
const sub_match<_Bi_iter> & __rhs ) [inline]
```

Tests the ordering of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` does not precede `__rhs`, false otherwise.

Definition at line 1222 of file `regex.h`.

```
3.67.3.43 template<typename _Bi_iter> bool std::operator>= ( const sub_match<_Bi_iter> & __lhs, typename
iterator_traits<_Bi_iter>::value_type const * __rhs ) [inline]
```

Tests the ordering of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A string.

Returns

true if `__lhs` does not precede `__rhs`, false otherwise.

Definition at line 1296 of file `regex.h`.

```
3.67.3.44 template<typename _Bi_iter> bool std::operator>= ( typename iterator_traits<_Bi_iter>::value_type const & __lhs,
const sub_match<_Bi_iter> & __rhs ) [inline]
```

Tests the ordering of a string and a regular expression submatch.

Parameters

<code>__lhs</code>	A string.
<code>__rhs</code>	A regular expression submatch.

Returns

true if `__lhs` does not precede `__rhs`, false otherwise.

Definition at line 1376 of file `regex.h`.

```
3.67.3.45 template<typename _Bi_iter > bool std::operator>= ( const sub_match<_Bi_iter > & __lhs, typename
iterator_traits<_Bi_iter >::value_type const & __rhs ) [inline]
```

Tests the ordering of a regular expression submatch and a string.

Parameters

<code>__lhs</code>	A regular expression submatch.
<code>__rhs</code>	A const string reference.

Returns

true if `__lhs` does not precede `__rhs`, false otherwise.

Definition at line 1456 of file `regex.h`.

```
3.67.3.46 template<typename _Bi_iter, typename _Alloc, typename _Ch_type, typename _Rx_traits > bool std::regex_match (
_Bi_iter __s, _Bi_iter __e, match_results<_Bi_iter, _Alloc > & __m, const basic_regex<_Ch_type, _Rx_traits > &
__re, regex_constants::match_flag_type __flags = regex_constants::match_default ) [inline]
```

Determines if there is a match between the regular expression `e` and all of the character sequence `[first, last)`.

Parameters

<code>__s</code>	Start of the character sequence to match.
<code>__e</code>	One-past-the-end of the character sequence to match.
<code>__m</code>	The match results.
<code>__re</code>	The regular expression.
<code>__flags</code>	Controls how the regular expression is matched.

Return values

<code>true</code>	A match exists.
<code>false</code>	Otherwise.

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

Definition at line 1988 of file `regex.h`.

Referenced by `std::regex_match()`.

```
3.67.3.47 template<typename _Bi_iter, typename _Ch_type, typename _Rx_traits> bool std::regex_match ( _Bi_iter __first,
    _Bi_iter __last, const basic_regex< _Ch_type, _Rx_traits> & __re, regex_constants::match_flag_type __flags
    = regex_constants::match_default ) [inline]
```

Indicates if there is a match between the regular expression `e` and all of the character sequence `[first, last)`.

Parameters

<code>__first</code>	Beginning of the character sequence to match.
<code>__last</code>	One-past-the-end of the character sequence to match.
<code>__re</code>	The regular expression.
<code>__flags</code>	Controls how the regular expression is matched.

Return values

<code>true</code>	A match exists.
<code>false</code>	Otherwise.

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

Definition at line 2016 of file `regex.h`.

References `std::regex_match()`.

```
3.67.3.48 template<typename _Ch_type, typename _Alloc, typename _Rx_traits> bool std::regex_match ( const _Ch_type *
    __s, match_results< const _Ch_type *, _Alloc> & __m, const basic_regex< _Ch_type, _Rx_traits> & __re,
    regex_constants::match_flag_type __f = regex_constants::match_default ) [inline]
```

Determines if there is a match between the regular expression `e` and a C-style null-terminated string.

Parameters

<code>__s</code>	The C-style null-terminated string to match.
<code>__m</code>	The match results.

Parameters

<code>_re</code>	The regular expression.
<code>_f</code>	Controls how the regular expression is matched.

Return values

<code>true</code>	A match exists.
<code>false</code>	Otherwise.

Exceptions

<code>an</code>	exception of type <code>regex_error</code> .
-----------------	--

Definition at line 2041 of file `regex.h`.

References `std::regex_match()`.

```
3.67.3.49 template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >
bool std::regex_match ( const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > & __s, match_results< typename
basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > & __m, const basic_regex< _Ch_type,
_Rx_traits > & __re, regex_constants::match_flag_type __flags = regex_constants::match_default )
[inline]
```

Determines if there is a match between the regular expression `e` and a string.

Parameters

<code>__s</code>	The string to match.
<code>__m</code>	The match results.
<code>__re</code>	The regular expression.
<code>__flags</code>	Controls how the regular expression is matched.

Return values

<code>true</code>	A match exists.
<code>false</code>	Otherwise.

Exceptions

<code>an</code>	exception of type <code>regex_error</code> .
-----------------	--

Definition at line 2065 of file `regex.h`.

References `std::basic_string< _CharT, _Traits, _Alloc >::begin()`, `std::basic_string< _CharT, _Traits, _Alloc >::end()`, `std::regex_constants::match_default`, and `std::regex_match()`.

```
3.67.3.50  template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >
            bool std::regex_match ( const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &&, match_results< typename
            basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &, const basic_regex< _Ch_type,
            _Rx_traits > &, regex_constants::match_flag_type = regex_constants::match_default ) [delete]
```

Prevent unsafe attempts to get `match_results` from a temporary string.

```
3.67.3.51  template<typename _Ch_type, class _Rx_traits > bool std::regex_match ( const _Ch_type* __s, const basic_regex<
            _Ch_type, _Rx_traits > & __re, regex_constants::match_flag_type __f = regex_constants::match_default )
            [inline]
```

Indicates if there is a match between the regular expression `e` and a C-style null-terminated string.

Parameters

<code>__s</code>	The C-style null-terminated string to match.
<code>__re</code>	The regular expression.
<code>__f</code>	Controls how the regular expression is matched.

Return values

<code>true</code>	A match exists.
<code>false</code>	Otherwise.

Exceptions

<code>an</code>	exception of type <code>regex_error</code> .
-----------------	--

Definition at line 2101 of file `regex.h`.

References `std::regex_match()`.

```
3.67.3.52  template<typename _Ch_traits, typename _Str_allocator, typename _Ch_type, typename _Rx_traits >
            bool std::regex_match ( const basic_string< _Ch_type, _Ch_traits, _Str_allocator > & __s, const
            basic_regex< _Ch_type, _Rx_traits > & __re, regex_constants::match_flag_type __flags =
            regex_constants::match_default ) [inline]
```

Indicates if there is a match between the regular expression `e` and a string.

Parameters

<code>__s</code>	[IN] The string to match.
<code>__re</code>	[IN] The regular expression.
<code>__flags</code>	[IN] Controls how the regular expression is matched.

Return values

<i>true</i>	A match exists.
<i>false</i>	Otherwise.

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

Definition at line 2123 of file `regex.h`.

References `std::basic_string<_CharT, _Traits, _Alloc>::begin()`, `std::basic_string<_CharT, _Traits, _Alloc>::end()`, and `std::regex_match()`.

3.67.3.53 `template<typename _Out_iter, typename _Bi_iter, typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa> _Out_iter std::regex_replace(_Out_iter __out, _Bi_iter __first, _Bi_iter __last, const basic_regex<_Ch_type, _Rx_traits> &__e, const basic_string<_Ch_type, _St, _Sa> &__fmt, regex_constants::match_flag_type __flags = regex_constants::match_default) [inline]`

Search for a regular expression within a range for multiple times, and replace the matched parts through filling a format string.

Parameters

<code>__out</code>	[OUT] The output iterator.
<code>__first</code>	[IN] The start of the string to search.
<code>__last</code>	[IN] One-past-the-end of the string to search.
<code>__e</code>	[IN] The regular expression to search for.
<code>__fmt</code>	[IN] The format string.
<code>__flags</code>	[IN] Search and replace policy flags.

Returns

`__out`

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

Definition at line 2294 of file `regex.h`.

References `std::basic_string<_CharT, _Traits, _Alloc>::c_str()`, and `std::regex_constants::match_default`.

Referenced by `std::regex_replace()`.

```

3.67.3.54 template<typename _Out_iter , typename _Bi_iter , typename _Rx_traits , typename _Ch_type > _Out_iter
std::regex_replace ( _Out_iter __out, _Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type, _Rx_traits > &
__e, const _Ch_type * __fmt, regex_constants::match_flag_type __flags = regex_constants::match_default
)

```

Search for a regular expression within a range for multiple times, and replace the matched parts through filling a format C-string.

Parameters

<code>__out</code>	[OUT] The output iterator.
<code>__first</code>	[IN] The start of the string to search.
<code>__last</code>	[IN] One-past-the-end of the string to search.
<code>__e</code>	[IN] The regular expression to search for.
<code>__fmt</code>	[IN] The format C-string.
<code>__flags</code>	[IN] Search and replace policy flags.

Returns

`__out`

Exceptions

<code>an</code>	exception of type <code>regex_error</code> .
-----------------	--

Definition at line 459 of file `regex.tcc`.

References `std::pair< _T1, _T2 >::first`, `std::regex_constants::format_first_only`, `std::regex_constants::format_no_copy`, `std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator==()`, and `std::pair< _T1, _T2 >::second`.

```

3.67.3.55 template<typename _Rx_traits , typename _Ch_type , typename _St , typename _Sa , typename _Fst , typename _Fsa
> basic_string< _Ch_type, _St, _Sa> std::regex_replace ( const basic_string< _Ch_type, _St, _Sa > & __s,
const basic_regex< _Ch_type, _Rx_traits > & __e, const basic_string< _Ch_type, _Fst, _Fsa > & __fmt,
regex_constants::match_flag_type __flags = regex_constants::match_default ) [inline]

```

Search for a regular expression within a string for multiple times, and replace the matched parts through filling a format string.

Parameters

<code>__s</code>	[IN] The string to search and replace.
<code>__e</code>	[IN] The regular expression to search for.
<code>__fmt</code>	[IN] The format string.
<code>__flags</code>	[IN] Search and replace policy flags.

Returns

The string after replacing.

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

Definition at line 2339 of file `regex.h`.

References `std::back_inserter()`, `std::basic_string<_CharT, _Traits, _Alloc >::begin()`, `std::basic_string<_CharT, _Traits, _Alloc >::end()`, and `std::regex_replace()`.

```
3.67.3.56 template<typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa > basic_string<_Ch_type,
    _St, _Sa> std::regex_replace ( const basic_string<_Ch_type, _St, _Sa > & __s, const basic_regex<
    _Ch_type, _Rx_traits > & __e, const _Ch_type * __fmt, regex_constants::match_flag_type __flags =
    regex_constants::match_default ) [inline]
```

Search for a regular expression within a string for multiple times, and replace the matched parts through filling a format C-string.

Parameters

<code>__s</code>	[IN] The string to search and replace.
<code>__e</code>	[IN] The regular expression to search for.
<code>__fmt</code>	[IN] The format C-string.
<code>__flags</code>	[IN] Search and replace policy flags.

Returns

The string after replacing.

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

Definition at line 2365 of file `regex.h`.

References `std::back_inserter()`, `std::basic_string<_CharT, _Traits, _Alloc >::begin()`, `std::basic_string<_CharT, _Traits, _Alloc >::end()`, and `std::regex_replace()`.

```
3.67.3.57 template<typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa > basic_string<_Ch_type>
    std::regex_replace ( const _Ch_type * __s, const basic_regex<_Ch_type, _Rx_traits > & __e, const basic_string<
    _Ch_type, _St, _Sa > & __fmt, regex_constants::match_flag_type __flags = regex_constants::match_default
    ) [inline]
```

Search for a regular expression within a C-string for multiple times, and replace the matched parts through filling a format string.

Parameters

<code>__s</code>	[IN] The C-string to search and replace.
<code>__e</code>	[IN] The regular expression to search for.
<code>__fmt</code>	[IN] The format string.
<code>__flags</code>	[IN] Search and replace policy flags.

Returns

The string after replacing.

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

Definition at line 2391 of file `regex.h`.

References `std::back_inserter()`, and `std::regex_replace()`.

```
3.67.3.58 template<typename _Rx_traits, typename _Ch_type> basic_string<_Ch_type> std::regex_replace
( const _Ch_type * __s, const basic_regex<_Ch_type, _Rx_traits> & __e, const _Ch_type * __fmt,
  regex_constants::match_flag_type __flags = regex_constants::match_default ) [inline]
```

Search for a regular expression within a C-string for multiple times, and replace the matched parts through filling a format C-string.

Parameters

<code>__s</code>	[IN] The C-string to search and replace.
<code>__e</code>	[IN] The regular expression to search for.
<code>__fmt</code>	[IN] The format C-string.
<code>__flags</code>	[IN] Search and replace policy flags.

Returns

The string after replacing.

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

Definition at line 2417 of file `regex.h`.

References `std::back_inserter()`, and `std::regex_replace()`.

```
3.67.3.59 template<typename _Bi_iter, typename _Alloc, typename _Ch_type, typename _Rx_traits> bool std::regex_search (
    _Bi_iter __s, _Bi_iter __e, match_results<_Bi_iter, _Alloc> & __m, const basic_regex<_Ch_type, _Rx_traits> &
    __re, regex_constants::match_flag_type __flags = regex_constants::match_default ) [inline]
```

Searches for a regular expression within a range.

Parameters

<code>__s</code>	[IN] The start of the string to search.
<code>__e</code>	[IN] One-past-the-end of the string to search.
<code>__m</code>	[OUT] The match results.
<code>__re</code>	[IN] The regular expression to search for.
<code>__flags</code>	[IN] Search policy flags.

Return values

<code>true</code>	A match was found within the string.
<code>false</code>	No match was found within the string, the content of <code>m</code> is undefined.

Exceptions

<code>an</code>	exception of type <code>regex_error</code> .
-----------------	--

Definition at line 2146 of file `regex.h`.

Referenced by `std::regex_iterator<_Bi_iter, _Ch_type, _Rx_traits>::operator++()`, `std::regex_iterator<_Bi_iter, _Ch_type, _Rx_traits>::regex_iterator()`, and `std::regex_search()`.

```
3.67.3.60 template<typename _Bi_iter, typename _Ch_type, typename _Rx_traits> bool std::regex_search ( _Bi_iter __first,
    _Bi_iter __last, const basic_regex<_Ch_type, _Rx_traits> & __re, regex_constants::match_flag_type __flags
    = regex_constants::match_default ) [inline]
```

Searches for a regular expression within a range.

Parameters

<code>__first</code>	[IN] The start of the string to search.
<code>__last</code>	[IN] One-past-the-end of the string to search.
<code>__re</code>	[IN] The regular expression to search for.
<code>__flags</code>	[IN] Search policy flags.

Return values

<code>true</code>	A match was found within the string.
<code>false</code>	No match was found within the string.

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

Definition at line 2170 of file `regex.h`.

References `std::regex_search()`.

```
3.67.3.61  template<typename _Ch_type, class _Alloc, class _Rx_traits > bool std::regex_search ( const _Ch_type * __s,
match_results< const _Ch_type *, _Alloc > & __m, const basic_regex< _Ch_type, _Rx_traits > & __e,
regex_constants::match_flag_type __f=regex_constants::match_default ) [inline]
```

Searches for a regular expression within a C-string.

Parameters

<code>__s</code>	[IN] A C-string to search for the regex.
<code>__m</code>	[OUT] The set of regex matches.
<code>__e</code>	[IN] The regex to search for in <code>s</code> .
<code>__f</code>	[IN] The search flags.

Return values

<code>true</code>	A match was found within the string.
<code>false</code>	No match was found within the string, the content of <code>m</code> is undefined.

Exceptions

<i>an</i>	exception of type <code>regex_error</code> .
-----------	--

Definition at line 2193 of file `regex.h`.

References `std::regex_search()`.

```
3.67.3.62  template<typename _Ch_type, typename _Rx_traits > bool std::regex_search ( const _Ch_type * __s,
const basic_regex< _Ch_type, _Rx_traits > & __e, regex_constants::match_flag_type __f =
regex_constants::match_default ) [inline]
```

Searches for a regular expression within a C-string.

Parameters

<code>__s</code>	[IN] The C-string to search.
<code>__e</code>	[IN] The regular expression to search for.
<code>__f</code>	[IN] Search policy flags.

Return values

<code>true</code>	A match was found within the string.
<code>false</code>	No match was found within the string.

Exceptions

<code>an</code>	exception of type <code>regex_error</code> .
-----------------	--

Definition at line 2212 of file `regex.h`.

References `std::regex_search()`.

```
3.67.3.63 template<typename _Ch_traits, typename _String_allocator, typename _Ch_type, typename _Rx_traits > bool
std::regex_search ( const basic_string< _Ch_type, _Ch_traits, _String_allocator > & __s, const basic_regex<
_Ch_type, _Rx_traits > & __e, regex_constants::match_flag_type __flags = regex_constants::match_default
) [inline]
```

Searches for a regular expression within a string.

Parameters

<code>__s</code>	[IN] The string to search.
<code>__e</code>	[IN] The regular expression to search for.
<code>__flags</code>	[IN] Search policy flags.

Return values

<code>true</code>	A match was found within the string.
<code>false</code>	No match was found within the string.

Exceptions

<code>an</code>	exception of type <code>regex_error</code> .
-----------------	--

Definition at line 2231 of file `regex.h`.

References `std::regex_search()`.

```
3.67.3.64  template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >
            bool std::regex_search ( const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &__s, match_results<
            typename basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &__m, const basic_regex<
            _Ch_type, _Rx_traits > &__e, regex_constants::match_flag_type __f=regex_constants::match_default )
            [inline]
```

Searches for a regular expression within a string.

Parameters

<code>__s</code>	[IN] A C++ string to search for the regex.
<code>__m</code>	[OUT] The set of regex matches.
<code>__e</code>	[IN] The regex to search for in <code>s</code> .
<code>__f</code>	[IN] The search flags.

Return values

<code>true</code>	A match was found within the string.
<code>false</code>	No match was found within the string, the content of <code>m</code> is undefined.

Exceptions

<code>an</code>	exception of type <code>regex_error</code> .
-----------------	--

Definition at line 2254 of file `regex.h`.

References `std::basic_string< _CharT, _Traits, _Alloc >::begin()`, `std::basic_string< _CharT, _Traits, _Alloc >::end()`, `std::regex_constants::match_default`, and `std::regex_search()`.

```
3.67.3.65  template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >
            bool std::regex_search ( const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &&, match_results< typename
            basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &, const basic_regex< _Ch_type,
            _Rx_traits > &, regex_constants::match_flag_type =regex_constants::match_default ) [delete]
```

Prevent unsafe attempts to get `match_results` from a temporary string.

```
3.67.3.66  template<typename _Ch_type, typename _Rx_traits > void std::swap ( basic_regex< _Ch_type, _Rx_traits > &__lhs,
            basic_regex< _Ch_type, _Rx_traits > &__rhs ) [inline]
```

Swaps the contents of two regular expression objects.

Parameters

<code>__lhs</code>	First regular expression.
<code>__rhs</code>	Second regular expression.

Definition at line 805 of file `regex.h`.

References `std::basic_regex<_Ch_type, _Rx_traits>::swap()`.

3.67.3.67 `template<typename _Bi_iter, typename _Alloc> void std::swap (match_results<_Bi_iter, _Alloc> &__lhs, match_results<_Bi_iter, _Alloc> &__rhs) [inline]`

Swaps two match results.

Parameters

<code>__lhs</code>	A match result.
<code>__rhs</code>	A match result.

The contents of the two `match_results` objects are swapped.

Definition at line 1958 of file `regex.h`.

References `std::match_results<_Bi_iter, _Alloc>::swap()`.

Referenced by `std::match_results<_Bi_iter>::swap()`.

3.68 SGI

Collaboration diagram for SGI:



Classes

- class `__gnu_cxx::binary_compose< _Operation1, _Operation2, _Operation3 >`
- struct `__gnu_cxx::constant_binary_fun< _Result, _Arg1, _Arg2 >`
- struct `__gnu_cxx::constant_unary_fun< _Result, _Argument >`
- struct `__gnu_cxx::constant_void_fun< _Result >`
- class `__gnu_cxx::hash_map< _Key, _Tp, _HashFn, _EqualKey, _Alloc >`
- class `__gnu_cxx::hash_multimap< _Key, _Tp, _HashFn, _EqualKey, _Alloc >`
- class `__gnu_cxx::hash_multiset< _Value, _HashFn, _EqualKey, _Alloc >`
- class `__gnu_cxx::hash_set< _Value, _HashFn, _EqualKey, _Alloc >`
- struct `__gnu_cxx::project1st< _Arg1, _Arg2 >`
- struct `__gnu_cxx::project2nd< _Arg1, _Arg2 >`
- struct `__gnu_cxx::rb_tree< _Key, _Value, _KeyOfValue, _Compare, _Alloc >`
- class `__gnu_cxx::rope< _CharT, _Alloc >`
- struct `__gnu_cxx::select1st< _Pair >`
- struct `__gnu_cxx::select2nd< _Pair >`
- class `__gnu_cxx::slist< _Tp, _Alloc >`
- class `__gnu_cxx::subtractive_rng`
- struct `__gnu_cxx::temporary_buffer< _ForwardIterator, _Tp >`
- class `__gnu_cxx::unary_compose< _Operation1, _Operation2 >`

Functions

- template<typename _Tp >
const _Tp & `__gnu_cxx::__median` (const _Tp &__a, const _Tp &__b, const _Tp &__c)
- template<typename _Tp, typename _Compare >
const _Tp & `__gnu_cxx::__median` (const _Tp &__a, const _Tp &__b, const _Tp &__c, _Compare __comp)
- size_t `std::bitset< _Nb >::__Find_first` () const noexcept
- size_t `std::bitset< _Nb >::__Find_next` (size_t __prev) const noexcept
- template<class _Operation1, class _Operation2 >
unary_compose< _Operation1, _Operation2 > `__gnu_cxx::compose1` (const _Operation1 &__fn1, const _Operation2 &__fn2)
- template<class _Operation1, class _Operation2, class _Operation3 >
binary_compose< _Operation1, _Operation2, _Operation3 > `__gnu_cxx::compose2` (const _Operation1 &__fn1, const _Operation2 &__fn2, const _Operation3 &__fn3)

- `template<class _Result >`
`constant_void_fun< _Result > __gnu_cxx::constant0 (const _Result &__val)`
 - `template<class _Result >`
`constant_unary_fun< _Result, _Result > __gnu_cxx::constant1 (const _Result &__val)`
 - `template<class _Result >`
`constant_binary_fun< _Result, _Result, _Result > __gnu_cxx::constant2 (const _Result &__val)`
 - `template<typename _InputIterator, typename _Size, typename _OutputIterator >`
`pair< _InputIterator, _OutputIterator > __gnu_cxx::copy_n (_InputIterator __first, _Size __count, _OutputIterator __result)`
 - `template<typename _InputIterator, typename _Distance >`
`void __gnu_cxx::distance (_InputIterator __first, _InputIterator __last, _Distance &__n)`
 - `template<class _Tp >`
`_Tp __gnu_cxx::identity_element (std::plus< _Tp >)`
 - `template<class _Tp >`
`_Tp __gnu_cxx::identity_element (std::multiplies< _Tp >)`
 - `template<typename _InputIterator1, typename _InputIterator2 >`
`int __gnu_cxx::lexicographical_compare_3way (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2)`
 - `template<typename _Tp, typename _Integer, typename _MonoidOperation >`
`_Tp __gnu_cxx::power (_Tp __x, _Integer __n, _MonoidOperation __monoid_op)`
 - `template<typename _Tp, typename _Integer >`
`_Tp __gnu_cxx::power (_Tp __x, _Integer __n)`
 - `template<typename _InputIterator, typename _RandomAccessIterator >`
`_RandomAccessIterator __gnu_cxx::random_sample (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out_first, _RandomAccessIterator __out_last)`
 - `template<typename _InputIterator, typename _RandomAccessIterator, typename _RandomNumberGenerator >`
`_RandomAccessIterator __gnu_cxx::random_sample (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out_first, _RandomAccessIterator __out_last, _RandomNumberGenerator &__rand)`
 - `template<typename _ForwardIterator, typename _OutputIterator, typename _Distance >`
`_OutputIterator __gnu_cxx::random_sample_n (_ForwardIterator __first, _ForwardIterator __last, _OutputIterator __out, const _Distance __n)`
 - `template<typename _ForwardIterator, typename _OutputIterator, typename _Distance, typename _RandomNumberGenerator >`
`_OutputIterator __gnu_cxx::random_sample_n (_ForwardIterator __first, _ForwardIterator __last, _OutputIterator __out, const _Distance __n, _RandomNumberGenerator &__rand)`
 - `template<typename _InputIter, typename _Size, typename _ForwardIter >`
`pair< _InputIter, _ForwardIter > __gnu_cxx::uninitialized_copy_n (_InputIter __first, _Size __count, _ForwardIter __result)`
-
- `bitset< _Nb > & std::bitset< _Nb >::Unchecked_set (size_t __pos) noexcept`
 - `bitset< _Nb > & std::bitset< _Nb >::Unchecked_set (size_t __pos, int __val) noexcept`
 - `bitset< _Nb > & std::bitset< _Nb >::Unchecked_reset (size_t __pos) noexcept`
 - `bitset< _Nb > & std::bitset< _Nb >::Unchecked_flip (size_t __pos) noexcept`
 - `constexpr bool std::bitset< _Nb >::Unchecked_test (size_t __pos) const noexcept`

3.68.1 Detailed Description

Because libstdc++ based its implementation of the STL subsections of the library on the SGI 3.3 implementation, we inherited their extensions as well.

They are additionally documented in the [online documentation](#), a copy of which is also shipped with the library source code (in `.../docs/html/documentation.html`). You can also read the documentation [on SGI's site](#), which is still running even though the code is not maintained.

NB that the following notes are pulled from various comments all over the place, so they may seem stilted.

The `identity_element` functions are not part of the C++ standard; SGI provided them as an extension. Its argument is an operation, and its return value is the identity element for that operation. It is overloaded for addition and multiplication, and you can overload it for your own nefarious operations.

As an extension to the binders, SGI provided composition functors and wrapper functions to aid in their creation. The `unary_compose` functor is constructed from two functions/functors, `f` and `g`. Calling `operator()` with a single argument `x` returns `f(g(x))`. The function `compose1` takes the two functions and constructs a `unary_compose` variable for you.

`binary_compose` is constructed from three functors, `f`, `g1`, and `g2`. Its `operator()` returns `f(g1(x),g2(x))`. The function `compose2` takes `f`, `g1`, and `g2`, and constructs the `binary_compose` instance for you. For example, if `f` returns an `int`, then

```
int answer = (compose2(f,g1,g2))(x);
```

is equivalent to

```
int temp1 = g1(x);
int temp2 = g2(x);
int answer = f(temp1,temp2);
```

But the first form is more compact, and can be passed around as a functor to other algorithms.

As an extension, SGI provided a functor called `identity`. When a functor is required but no operations are desired, this can be used as a pass-through. Its `operator()` returns its argument unchanged.

`select1st` and `select2nd` are extensions provided by SGI. Their `operator()`s take a `std::pair` as an argument, and return either the first member or the second member, respectively. They can be used (especially with the composition functors) to *strip* data from a sequence before performing the remainder of an algorithm.

The `operator()` of the `project1st` functor takes two arbitrary arguments and returns the first one, while `project2nd` returns the second one. They are extensions provided by SGI.

These three functors are each constructed from a single arbitrary variable/value. Later, their `operator()`s completely ignore any arguments passed, and return the stored value.

- `constant_void_fun`'s `operator()` takes no arguments
- `constant_unary_fun`'s `operator()` takes one argument (ignored)
- `constant_binary_fun`'s `operator()` takes two arguments (ignored)

The helper creator functions `constant0`, `constant1`, and `constant2` each take a *result* argument and construct variables of the appropriate functor type.

3.68.2 Function Documentation

3.68.2.1 `template<typename _Tp> const _Tp& __gnu_cxx::__median (const _Tp & __a, const _Tp & __b, const _Tp & __c)`

Find the median of three values.

Parameters

\leftrightarrow _a	A value.
\leftrightarrow _b	A value.
\leftrightarrow _c	A value.

Returns

One of a, b or c.

If $\{1,m,n\}$ is some convolution of $\{a,b,c\}$ such that $1 \leq m \leq n$ then the value returned will be m. This is an SGI extension.

Definition at line 546 of file ext/algorithm.

3.68.2.2 `template<typename _Tp, typename _Compare> const _Tp& __gnu_cxx::__median (const _Tp & __a, const _Tp & __b, const _Tp & __c, _Compare __comp)`

Find the median of three values using a predicate for comparison.

Parameters

__a	A value.
__b	A value.
__c	A value.
__comp	A binary predicate.

Returns

One of a, b or c.

If $\{1,m,n\}$ is some convolution of $\{a,b,c\}$ such that `comp (1, m)` and `comp (m, n)` are both true then the value returned will be m. This is an SGI extension.

Definition at line 580 of file ext/algorithm.

3.68.2.3 `template<size_t _Nb> size_t std::bitset<_Nb>::_Find_first () const [inline], [noexcept]`

Finds the index of the first "on" bit.

Returns

The index of the first bit set, or size() if not found.

See also

`_Find_next`

Definition at line 1363 of file bitset.

3.68.2.4 `template<size_t _Nb> size_t std::bitset<_Nb>::_Find_next (size_t __prev) const` `[inline], [noexcept]`

Finds the index of the next "on" bit after prev.

Returns

The index of the next bit set, or size() if not found.

Parameters

<code>__prev</code>	Where to start searching.
---------------------	---------------------------

See also

`_Find_first`

Definition at line 1374 of file `bitset`.

3.68.2.5 `template<size_t _Nb> bitset<_Nb>& std::bitset<_Nb>::_Unchecked_flip (size_t __pos)` `[inline], [noexcept]`

These versions of single-bit set, reset, flip, and test are extensions from the SGI version. They do no range checking.

Definition at line 1050 of file `bitset`.

3.68.2.6 `template<size_t _Nb> bitset<_Nb>& std::bitset<_Nb>::_Unchecked_reset (size_t __pos)` `[inline], [noexcept]`

These versions of single-bit set, reset, flip, and test are extensions from the SGI version. They do no range checking.

Definition at line 1043 of file `bitset`.

3.68.2.7 `template<size_t _Nb> bitset<_Nb>& std::bitset<_Nb>::_Unchecked_set (size_t __pos)` `[inline], [noexcept]`

These versions of single-bit set, reset, flip, and test are extensions from the SGI version. They do no range checking.

Definition at line 1026 of file `bitset`.

3.68.2.8 `template<size_t _Nb> bitset<_Nb>& std::bitset<_Nb>::_Unchecked_set (size_t __pos, int __val)` `[inline], [noexcept]`

These versions of single-bit set, reset, flip, and test are extensions from the SGI version. They do no range checking.

Definition at line 1033 of file `bitset`.

```
3.68.2.9  template<size_t _Nb> constexpr bool std::bitset<_Nb>::_Unchecked_test ( size_t __pos ) const  [inline],
        [noexcept]
```

These versions of single-bit set, reset, flip, and test are extensions from the SGI version. They do no range checking.

Definition at line 1057 of file `bitset`.

```
3.68.2.10  template<class _Operation1 , class _Operation2 > unary_compose<_Operation1, _Operation2> __gnu_cxx::compose1 (
        const _Operation1 & __fn1, const _Operation2 & __fn2 )  [inline]
```

An [SGI extension](#) .

Definition at line 145 of file `ext/functional`.

```
3.68.2.11  template<class _Operation1 , class _Operation2 , class _Operation3 > binary_compose<_Operation1, _Operation2,
        _Operation3> __gnu_cxx::compose2 ( const _Operation1 & __fn1, const _Operation2 & __fn2, const _Operation3 &
        __fn3 )  [inline]
```

An [SGI extension](#) .

Definition at line 172 of file `ext/functional`.

```
3.68.2.12  template<class _Result > constant_void_fun<_Result> __gnu_cxx::constant0 ( const _Result & __val )  [inline]
```

An [SGI extension](#) .

Definition at line 330 of file `ext/functional`.

```
3.68.2.13  template<class _Result > constant_unary_fun<_Result, _Result> __gnu_cxx::constant1 ( const _Result & __val )
        [inline]
```

An [SGI extension](#) .

Definition at line 336 of file `ext/functional`.

```
3.68.2.14  template<class _Result > constant_binary_fun<_Result, _Result, _Result> __gnu_cxx::constant2 ( const _Result &
        __val )  [inline]
```

An [SGI extension](#) .

Definition at line 342 of file `ext/functional`.

```
3.68.2.15  template<typename _InputIterator , typename _Size , typename _OutputIterator > pair<_InputIterator, _OutputIterator>
        __gnu_cxx::copy_n ( _InputIterator __first, _Size __count, _OutputIterator __result )  [inline]
```

Copies the range `[first,first+count)` into `[result,result+count)`.

Parameters

<code>__first</code>	An input iterator.
<code>__count</code>	The number of elements to copy.
<code>__result</code>	An output iterator.

Returns

A `std::pair` composed of `first+count` and `result+count`.

This is an SGI extension. This inline function will boil down to a call to `memmove` whenever possible. Failing that, if random access iterators are passed, then the loop count will be known (and therefore a candidate for compiler optimizations such as unrolling).

Definition at line 120 of file `ext/algorithm`.

```
3.68.2.16  template<typename _InputIterator, typename _Distance> void __gnu_cxx::distance ( _InputIterator __first,
        _InputIterator __last, _Distance &__n )  [inline]
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation_style.html

Definition at line 105 of file `ext/iterator`.

```
3.68.2.17  template<class _Tp> _Tp __gnu_cxx::identity_element ( std::plus<_Tp> )  [inline]
```

An [SGI extension](#) .

Definition at line 87 of file `ext/functional`.

```
3.68.2.18  template<class _Tp> _Tp __gnu_cxx::identity_element ( std::multiplies<_Tp> )  [inline]
```

An [SGI extension](#) .

Definition at line 93 of file `ext/functional`.

```
3.68.2.19  template<typename _InputIterator1, typename _InputIterator2> int __gnu_cxx::lexicographical_compare_3way (
        _InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2 )
```

`memcmp` on steroids.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.
<code>__last2</code>	An input iterator.

Returns

An int, as with `memcmp`.

The return value will be less than zero if the first range is *lexigraphically less than* the second, greater than zero if the second range is *lexigraphically less than* the first, and zero otherwise. This is an SGI extension.

Definition at line 201 of file `ext/algorithm`.

```
3.68.2.20 template<typename _Tp, typename _Integer, typename _MonoidOperation > _Tp __gnu_cxx::power ( _Tp __x, _Integer
    __n, _MonoidOperation __monoid_op ) [inline]
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

Definition at line 113 of file `ext/numeric`.

```
3.68.2.21 template<typename _Tp, typename _Integer > _Tp __gnu_cxx::power ( _Tp __x, _Integer __n ) [inline]
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

Definition at line 123 of file `ext/numeric`.

```
3.68.2.22 template<typename _InputIterator, typename _RandomAccessIterator > _RandomAccessIterator
    __gnu_cxx::random_sample ( _InputIterator __first, _InputIterator __last, _RandomAccessIterator __out_first,
    _RandomAccessIterator __out_last ) [inline]
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

Definition at line 388 of file `ext/algorithm`.

```
3.68.2.23 template<typename _InputIterator, typename _RandomAccessIterator, typename _RandomNumberGenerator >
    _RandomAccessIterator __gnu_cxx::random_sample ( _InputIterator __first, _InputIterator __last, _RandomAccessIterator
    __out_first, _RandomAccessIterator __out_last, _RandomNumberGenerator & __rand ) [inline]
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

Definition at line 411 of file `ext/algorithm`.

```
3.68.2.24  template<typename _ForwardIterator , typename _OutputIterator , typename _Distance > _OutputIterator
           __gnu_cxx::random_sample_n ( _ForwardIterator __first, _ForwardIterator __last, _OutputIterator __out, const _Distance
           __n )
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

Definition at line 267 of file ext/algorithm.

```
3.68.2.25  template<typename _ForwardIterator , typename _OutputIterator , typename _Distance , typename
           _RandomNumberGenerator > _OutputIterator __gnu_cxx::random_sample_n ( _ForwardIterator __first, _ForwardIterator
           __last, _OutputIterator __out, const _Distance __n, _RandomNumberGenerator & __rand )
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

Definition at line 301 of file ext/algorithm.

```
3.68.2.26  template<typename _InputIter , typename _Size , typename _ForwardIter > pair<_InputIter, _ForwardIter>
           __gnu_cxx::uninitialized_copy_n ( _InputIter __first, _Size __count, _ForwardIter __result ) [inline]
```

Copies the range [first,last) into result.

Parameters

<code>__first</code>	An input iterator.
<code>__count</code>	Length
<code>__result</code>	An output iterator.

Returns

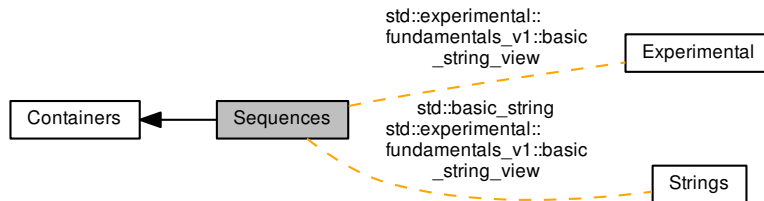
`__result + (__first + __count)`

Like `copy()`, but does not require an initialized output range.

Definition at line 122 of file ext/memory.

3.69 Sequences

Collaboration diagram for Sequences:



Classes

- struct `std::array<_Tp, _Nm>`
- class `std::basic_string<_CharT, _Traits, _Alloc>`
- class `std::deque<_Tp, _Alloc>`
- class `std::experimental::fundamentals_v1::basic_string_view<_CharT, _Traits>`
- class `std::forward_list<_Tp, _Alloc>`
- class `std::list<_Tp, _Alloc>`
- class `std::priority_queue<_Tp, _Sequence, _Compare>`
- class `std::queue<_Tp, _Sequence>`
- class `std::stack<_Tp, _Sequence>`
- class `std::vector<_Tp, _Alloc>`
- class `std::vector<bool, _Alloc>`

3.69.1 Detailed Description

Sequences arrange a collection of objects into a strictly linear order.

The differences between sequences are usually due to one or both of the following:

- memory management
- algorithmic complexity

As an example of the first case, `vector` is required to use a contiguous memory layout, while other sequences such as `deque` are not.

The prime reason for choosing one sequence over another should be based on the second category of differences, algorithmic complexity. For example, if you need to perform many inserts and removals from the middle of a sequence, `list` would be ideal. But if you need to perform constant-time access to random elements of the sequence, then `list` should not be used.

All sequences must meet certain requirements, summarized in [tables](#).

3.70 Set Operation

Collaboration diagram for Set Operation:



Functions

- `template<typename _InputIterator1, typename _InputIterator2 >`
`bool std::includes (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Compare >`
`bool std::includes (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`_OutputIterator std::set_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::set_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`_OutputIterator std::set_intersection (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::set_intersection (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`_OutputIterator std::set_symmetric_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::set_symmetric_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`_OutputIterator std::set_union (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::set_union (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`

3.70.1 Detailed Description

These algorithms are common set operations performed on sequences that are already sorted. The number of comparisons will be linear.

3.70.2 Function Documentation

3.70.2.1 `template<typename _InputIterator1, typename _InputIterator2 > bool std::includes (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2) [inline]`

Determines whether all elements of a sequence exists in a range.

Parameters

<code>__first1</code>	Start of search range.
<code>__last1</code>	End of search range.
<code>__first2</code>	Start of sequence
<code>__last2</code>	End of sequence.

Returns

True if each element in `[__first2,__last2)` is contained in order within `[__first1,__last1)`. False otherwise.

This operation expects both `[__first1,__last1)` and `[__first2,__last2)` to be sorted. Searches for the presence of each element in `[__first2,__last2)` within `[__first1,__last1)`. The iterators over each range only move forward, so this is a linear algorithm. If an element in `[__first2,__last2)` is not found before the search iterator reaches `__last2`, false is returned.

Definition at line 2823 of file `stl_algo.h`.

3.70.2.2 `template<typename _InputIterator1, typename _InputIterator2, typename _Compare > bool std::includes (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _Compare __comp) [inline]`

Determines whether all elements of a sequence exists in a range using comparison.

Parameters

<code>__first1</code>	Start of search range.
<code>__last1</code>	End of search range.
<code>__first2</code>	Start of sequence
<code>__last2</code>	End of sequence.
<code>__comp</code>	Comparison function to use.

Returns

True if each element in `[__first2,__last2)` is contained in order within `[__first1,__last1)` according to `comp`. False otherwise.

This operation expects both `[__first1,__last1)` and `[__first2,__last2)` to be sorted. Searches for the presence of each element in `[__first2,__last2)` within `[__first1,__last1)`, using `comp` to decide. The iterators over each range only move forward, so this is a linear algorithm. If an element in `[__first2,__last2)` is not found before the search iterator reaches `__last2`, false is returned.

Definition at line 2868 of file `stl_algo.h`.

References `std::__iterator_category()`, `std::__reverse()`, and `std::iter_swap()`.


```
3.70.2.3 template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator > _OutputIterator
std::set_difference ( _InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2,
    _OutputIterator __result ) [inline]
```

Return the difference of two sorted ranges.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__last2</code>	End of second range.

Returns

End of the output range.

This operation iterates over both ranges, copying elements present in the first range but not the second in order to the output range. Iterators increment for each range. When the current element of the first range is less than the second, that element is copied and the iterator advances. If the current element of the second range is less, the iterator advances, but no element is copied. If an element is contained in both ranges, no elements are copied and both ranges advance. The output range may not overlap either input range.

Definition at line 5244 of file `stl_algo.h`.

```
3.70.2.4 template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare
    > _OutputIterator std::set_difference ( _InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2,
    _InputIterator2 __last2, _OutputIterator __result, _Compare __comp ) [inline]
```

Return the difference of two sorted ranges using comparison functor.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__last2</code>	End of second range.
<code>__comp</code>	The comparison functor.

Returns

End of the output range.

This operation iterates over both ranges, copying elements present in the first range but not the second in order to the output range. Iterators increment for each range. When the current element of the first range is less than the second according to `__comp`, that element is copied and the iterator advances. If the current element of the second range is less, no element is copied and the iterator advances. If an element is contained in both ranges according to `__comp`, no elements are copied and both ranges advance. The output range may not overlap either input range.

Definition at line 5294 of file `stl_algo.h`.

```
3.70.2.5 template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator > _OutputIterator
std::set_intersection ( _InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2,
    _OutputIterator __result ) [inline]
```

Return the intersection of two sorted ranges.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__last2</code>	End of second range.

Returns

End of the output range.

This operation iterates over both ranges, copying elements present in both ranges in order to the output range. Iterators increment for each range. When the current element of one range is less than the other, that iterator advances. If an element is contained in both ranges, the element from the first range is copied and both ranges advance. The output range may not overlap either input range.

Definition at line 5124 of file `stl_algo.h`.

```
3.70.2.6 template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare
    > _OutputIterator std::set_intersection ( _InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2,
        _InputIterator2 __last2, _OutputIterator __result, _Compare __comp ) [inline]
```

Return the intersection of two sorted ranges using comparison functor.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__last2</code>	End of second range.
<code>__comp</code>	The comparison functor.

Returns

End of the output range.

This operation iterates over both ranges, copying elements present in both ranges in order to the output range. Iterators increment for each range. When the current element of one range is less than the other according to `__comp`, that iterator advances. If an element is contained in both ranges according to `__comp`, the element from the first range is copied and both ranges advance. The output range may not overlap either input range.

Definition at line 5172 of file `stl_algo.h`.

```
3.70.2.7 template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator > _OutputIterator
std::set_symmetric_difference ( _InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2
__last2, _OutputIterator __result ) [inline]
```

Return the symmetric difference of two sorted ranges.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__last2</code>	End of second range.

Returns

End of the output range.

This operation iterates over both ranges, copying elements present in one range but not the other in order to the output range. Iterators increment for each range. When the current element of one range is less than the other, that element is copied and the iterator advances. If an element is contained in both ranges, no elements are copied and both ranges advance. The output range may not overlap either input range.

Definition at line 5372 of file `stl_algo.h`.

```
3.70.2.8 template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >
_OutputIterator std::set_symmetric_difference ( _InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2,
_InputIterator2 __last2, _OutputIterator __result, _Compare __comp ) [inline]
```

Return the symmetric difference of two sorted ranges using comparison functor.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__last2</code>	End of second range.
<code>__comp</code>	The comparison functor.

Returns

End of the output range.

This operation iterates over both ranges, copying elements present in one range but not the other in order to the output range. Iterators increment for each range. When the current element of one range is less than the other according to `comp`, that element is copied and the iterator advances. If an element is contained in both ranges according to `__comp`, no elements are copied and both ranges advance. The output range may not overlap either input range.

Definition at line 5422 of file `stl_algo.h`.

```
3.70.2.9 template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator > _OutputIterator std::set_union
( _InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result
) [inline]
```

Return the union of two sorted ranges.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__last2</code>	End of second range.

Returns

End of the output range.

This operation iterates over both ranges, copying elements present in each range in order to the output range. Iterators increment for each range. When the current element of one range is less than the other, that element is copied and the iterator advanced. If an element is contained in both ranges, the element from the first range is copied and both ranges advance. The output range may not overlap either input range.

Definition at line 5005 of file `stl_algo.h`.

```
3.70.2.10 template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >
_OutputIterator std::set_union ( _InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2
__last2, _OutputIterator __result, _Compare __comp ) [inline]
```

Return the union of two sorted ranges using a comparison functor.

Parameters

<code>__first1</code>	Start of first range.
<code>__last1</code>	End of first range.
<code>__first2</code>	Start of second range.
<code>__last2</code>	End of second range.
<code>__comp</code>	The comparison functor.

Returns

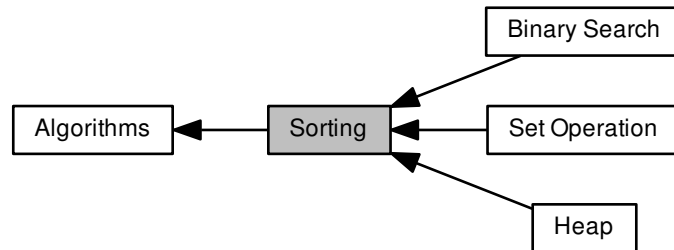
End of the output range.

This operation iterates over both ranges, copying elements present in each range in order to the output range. Iterators increment for each range. When the current element of one range is less than the other according to `__comp`, that element is copied and the iterator advanced. If an equivalent element according to `__comp` is contained in both ranges, the element from the first range is copied and both ranges advance. The output range may not overlap either input range.

Definition at line 5054 of file `stl_algo.h`.

3.71 Sorting

Collaboration diagram for Sorting:



Modules

- [Binary Search](#)
- [Heap](#)
- [Set Operation](#)

Functions

- `template<typename _BidirectionalIterator >`
`void std::inplace_merge (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`void std::inplace_merge (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last, _Compare __comp)`
- `template<typename _ForwardIterator >`
`bool std::is_sorted (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`bool std::is_sorted (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _ForwardIterator >`
`_ForwardIterator std::is_sorted_until (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`_ForwardIterator std::is_sorted_until (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _II1, typename _II2 >`
`bool std::lexicographical_compare (_II1 __first1, _II1 __last1, _II2 __first2, _II2 __last2)`
- `template<typename _II1, typename _II2, typename _Compare >`
`bool std::lexicographical_compare (_II1 __first1, _II1 __last1, _II2 __first2, _II2 __last2, _Compare __comp)`
- `template<typename _Tp >`
`_GLIBCXX14_CONSTEXPR const _Tp & std::max (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp, typename _Compare >`
`_GLIBCXX14_CONSTEXPR const _Tp & std::max (const _Tp &__a, const _Tp &__b, _Compare __comp)`

- `template<typename _ForwardIterator >`
`_GLIBCXX14_CONSTEXPR _ForwardIterator std::max_element (_ForwardIterator __first, _ForwardIterator __↵`
`last)`
- `template<typename _ForwardIterator, typename _Compare >`
`_GLIBCXX14_CONSTEXPR _ForwardIterator std::max_element (_ForwardIterator __first, _ForwardIterator __↵`
`last, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`_OutputIterator std::merge (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Input↵`
`Iterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::merge (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Input↵`
`Iterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _Tp >`
`_GLIBCXX14_CONSTEXPR const _Tp & std::min (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp, typename _Compare >`
`_GLIBCXX14_CONSTEXPR const _Tp & std::min (const _Tp &__a, const _Tp &__b, _Compare __comp)`
- `template<typename _ForwardIterator >`
`_GLIBCXX14_CONSTEXPR _ForwardIterator std::min_element (_ForwardIterator __first, _ForwardIterator __↵`
`last)`
- `template<typename _ForwardIterator, typename _Compare >`
`_GLIBCXX14_CONSTEXPR _ForwardIterator std::min_element (_ForwardIterator __first, _ForwardIterator __↵`
`last, _Compare __comp)`
- `template<typename _Tp >`
`_GLIBCXX14_CONSTEXPR pair< const _Tp &, const _Tp & > std::minmax (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp, typename _Compare >`
`_GLIBCXX14_CONSTEXPR pair< const _Tp &, const _Tp & > std::minmax (const _Tp &__a, const _Tp &__b,`
`_Compare __comp)`
- `template<typename _ForwardIterator >`
`_GLIBCXX14_CONSTEXPR pair< _ForwardIterator, _ForwardIterator > std::minmax_element (_ForwardIterator`
`__first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`_GLIBCXX14_CONSTEXPR pair< _ForwardIterator, _ForwardIterator > std::minmax_element (_ForwardIterator`
`__first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _BidirectionalIterator >`
`bool std::next_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`bool std::next_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`void std::nth_element (_RandomAccessIterator __first, _RandomAccessIterator __nth, _RandomAccessIterator`
`__last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::nth_element (_RandomAccessIterator __first, _RandomAccessIterator __nth, _RandomAccessIterator`
`__last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`void std::partial_sort (_RandomAccessIterator __first, _RandomAccessIterator __middle, _RandomAccess↵`
`Iterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::partial_sort (_RandomAccessIterator __first, _RandomAccessIterator __middle, _RandomAccess↵`
`Iterator __last, _Compare __comp)`
- `template<typename _InputIterator, typename _RandomAccessIterator >`
`_RandomAccessIterator std::partial_sort_copy (_InputIterator __first, _InputIterator __last, _RandomAccess↵`
`Iterator __result_first, _RandomAccessIterator __result_last)`

- `template<typename _InputIterator, typename _RandomAccessIterator, typename _Compare >`
`_RandomAccessIterator std::partial_sort_copy (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __result_first, _RandomAccessIterator __result_last, _Compare __comp)`
- `template<typename _BidirectionalIterator >`
`bool std::prev_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`bool std::prev_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`void std::sort (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`void std::stable_sort (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::stable_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`

3.71.1 Detailed Description

3.71.2 Function Documentation

3.71.2.1 `template<typename _BidirectionalIterator > void std::inplace_merge (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last) [inline]`

Merges two sorted ranges in place.

Parameters

<code>__first</code>	An iterator.
<code>__middle</code>	Another iterator.
<code>__last</code>	Another iterator.

Returns

Nothing.

Merges two sorted and consecutive ranges, `[__first, __middle)` and `[__middle, __last)`, and puts the result in `[__first, __last)`. The output will be sorted. The sort is *stable*, that is, for equivalent elements in the two ranges, elements from the first range will always come before elements from the second.

If enough additional memory is available, this takes $(\text{__last} - \text{__first}) - 1$ comparisons. Otherwise an $N \log N$ algorithm is used, where N is `distance(__first, __last)`.

Definition at line 2571 of file `stl_algo.h`.

3.71.2.2 `template<typename _BidirectionalIterator, typename _Compare > void std::inplace_merge (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last, _Compare __comp) [inline]`

Merges two sorted ranges in place.

Parameters

<code>__first</code>	An iterator.
<code>__middle</code>	Another iterator.
<code>__last</code>	Another iterator.
<code>__comp</code>	A functor to use for comparisons.

Returns

Nothing.

Merges two sorted and consecutive ranges, `[__first,__middle)` and `[middle,last)`, and puts the result in `[__first,__last)`. The output will be sorted. The sort is *stable*, that is, for equivalent elements in the two ranges, elements from the first range will always come before elements from the second.

If enough additional memory is available, this takes $(_last_first)-1$ comparisons. Otherwise an $N\log N$ algorithm is used, where N is `distance(__first,__last)`.

The comparison function should have the same effects on ordering as the function used for the initial sort.

Definition at line 2612 of file `stl_algo.h`.

3.71.2.3 `template<typename _ForwardIterator> bool std::is_sorted (_ForwardIterator __first, _ForwardIterator __last)`
`[inline]`

Determines whether the elements of a sequence are sorted.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.

Returns

True if the elements are sorted, false otherwise.

Definition at line 3206 of file `stl_algo.h`.

References `std::is_sorted_until()`.

3.71.2.4 `template<typename _ForwardIterator, typename _Compare> bool std::is_sorted (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)` `[inline]`

Determines whether the elements of a sequence are sorted according to a comparison functor.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__comp</code>	A comparison functor.

Returns

True if the elements are sorted, false otherwise.

Definition at line 3220 of file `stl_algo.h`.

References `std::is_sorted_until()`.

3.71.2.5 `template<typename _ForwardIterator> _ForwardIterator std::is_sorted_until (_ForwardIterator __first, _ForwardIterator __last) [inline]`

Determines the end of a sorted sequence.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.

Returns

An iterator pointing to the last iterator `i` in `[__first, __last)` for which the range `[__first, i)` is sorted.

Definition at line 3249 of file `stl_algo.h`.

3.71.2.6 `template<typename _ForwardIterator, typename _Compare> _ForwardIterator std::is_sorted_until (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp) [inline]`

Determines the end of a sorted sequence using comparison functor.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__comp</code>	A comparison functor.

Returns

An iterator pointing to the last iterator `i` in `[__first, __last)` for which the range `[__first, i)` is sorted.

Definition at line 3273 of file `stl_algo.h`.

Referenced by `std::is_sorted()`.

3.71.2.7 `template<typename _II1, typename _II2> bool std::lexicographical_compare (_II1 __first1, _II1 __last1, _II2 __first2, _II2 __last2) [inline]`

Performs **dictionary** comparison on ranges.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.
<code>__last2</code>	An input iterator.

Returns

A boolean true or false.

Returns true if the sequence of elements defined by the range `[first1,last1)` is lexicographically less than the sequence of elements defined by the range `[first2,last2)`. Returns false otherwise. (Quoted from [25.3.8]/1.) If the iterators are all character pointers, then this is an inline call to `memcmp`.

Definition at line 1203 of file `stl_algobase.h`.

```
3.71.2.8  template<typename _I1, typename _I2, typename _Compare > bool std::lexicographical_compare ( _I1 __first1, _I1
        __last1, _I2 __first2, _I2 __last2, _Compare __comp ) [inline]
```

Performs **dictionary** comparison on ranges.

Parameters

<code>__first1</code>	An input iterator.
<code>__last1</code>	An input iterator.
<code>__first2</code>	An input iterator.
<code>__last2</code>	An input iterator.
<code>__comp</code>	A comparison functor .

Returns

A boolean true or false.

The same as the four-parameter `lexicographical_compare`, but uses the `comp` parameter instead of `<`.

Definition at line 1239 of file `stl_algobase.h`.

Referenced by `std::operator<()`.

```
3.71.2.9  template<typename _Tp > _GLIBCXX14_CONSTEXPR const _Tp & std::max ( const _Tp & __a, const _Tp & __b )
        [inline]
```

This does what you think it does.

Parameters

<code>_↔ _a</code>	A thing of arbitrary type.
<code>_↔ _b</code>	Another thing of arbitrary type.

Returns

The greater of the parameters.

This is the simple classic generic implementation. It will work on temporary expressions, since they are only evaluated once, unlike a preprocessor macro.

Definition at line 219 of file `stl_algobase.h`.

Referenced by `__gnu_parallel::__parallel_nth_element()`, `std::vector< sub_match< _Bi_iter >, allocator< sub_match< _Bi_iter > > >::__M_allocate_and_copy()`, `std::Deque_base< _Tp, _Alloc >::__M_initialize_map()`, `std::deque< _Tp, _Alloc >::__M_reallocate_map()`, `std::discard_block_engine< _RandomNumberEngine, __p, __r >::__max()`, `std::shuffle_order_engine< _RandomNumberEngine, __k >::__max()`, `std::minmax_element()`, `__gnu_parallel::__multiseq_partition()`, `__gnu_parallel::multiseq_selection()`, `std::operator>>()`, and `std::basic_stringbuf< _CharT, __Traits, _Alloc >::overflow()`.

```
3.71.2.10 template<typename _Tp, typename _Compare > _GLIBCXX14_CONSTEXPR const _Tp & std::max ( const _Tp & __a,
const _Tp & __b, _Compare __comp ) [inline]
```

This does what you think it does.

Parameters

<code>__a</code>	A thing of arbitrary type.
<code>__b</code>	Another thing of arbitrary type.
<code>__comp</code>	A comparison functor .

Returns

The greater of the parameters.

This will work on temporary expressions, since they are only evaluated once, unlike a preprocessor macro.

Definition at line 265 of file `stl_algobase.h`.

```
3.71.2.11 template<typename _ForwardIterator > _GLIBCXX14_CONSTEXPR _ForwardIterator std::max_element ( _ForwardIterator
__first, _ForwardIterator __last ) [inline]
```

Return the maximum element in a range.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.

Returns

Iterator referencing the first instance of the largest value.

Definition at line 5539 of file `stl_algo.h`.

```
3.71.2.12 template<typename _ForwardIterator, typename _Compare> _GLIBCXX14_CONSTEXPR _ForwardIterator
std::max_element ( _ForwardIterator __first, _ForwardIterator __last, _Compare __comp ) [inline]
```

Return the maximum element in a range using comparison functor.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__comp</code>	Comparison functor.

Returns

Iterator referencing the first instance of the largest value according to `__comp`.

Definition at line 5564 of file `stl_algo.h`.

Referenced by `std::minmax_element()`.

```
3.71.2.13 template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator> _OutputIterator std::merge (
    _InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result )
[inline]
```

Merges two sorted ranges.

Parameters

<code>__first1</code>	An iterator.
<code>__first2</code>	Another iterator.
<code>__last1</code>	Another iterator.
<code>__last2</code>	Another iterator.
<code>__result</code>	An iterator pointing to the end of the merged range.

Returns

An iterator pointing to the first element *not less than* *val*.

Merges the ranges `[__first1, __last1)` and `[__first2, __last2)` into the sorted range `[__result, __result + (__last1 - __first1) + (__last2 - __first2))`. Both input ranges must be sorted, and the output range must not overlap with either of the input ranges. The sort is *stable*, that is, for equivalent elements in the two ranges, elements from the first range will always come before elements from the second.

Definition at line 4789 of file `stl_algo.h`.

```
3.71.2.14  template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >
            _OutputIterator std::merge ( _InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2
            __last2, _OutputIterator __result, _Compare __comp ) [inline]
```

Merges two sorted ranges.

Parameters

<code>__first1</code>	An iterator.
<code>__first2</code>	Another iterator.
<code>__last1</code>	Another iterator.
<code>__last2</code>	Another iterator.
<code>__result</code>	An iterator pointing to the end of the merged range.
<code>__comp</code>	A functor to use for comparisons.

Returns

An iterator pointing to the first element "not less than" *val*.

Merges the ranges `[__first1, __last1)` and `[__first2, __last2)` into the sorted range `[__result, __result + (__last1 - __first1) + (__last2 - __first2))`. Both input ranges must be sorted, and the output range must not overlap with either of the input ranges. The sort is *stable*, that is, for equivalent elements in the two ranges, elements from the first range will always come before elements from the second.

The comparison function should have the same effects on ordering as the function used for the initial sort.

Definition at line 4839 of file `stl_algo.h`.

References `std::__inplace_stable_sort()`.

```
3.71.2.15  template<typename _Tp > _GLIBCXX14_CONSTEXPR const _Tp & std::min ( const _Tp & __a, const _Tp & __b )
            [inline]
```

This does what you think it does.

Parameters

<code>__a</code>	A thing of arbitrary type.
<code>__b</code>	Another thing of arbitrary type.

Returns

The lesser of the parameters.

This is the simple classic generic implementation. It will work on temporary expressions, since they are only evaluated once, unlike a preprocessor macro.

Definition at line 195 of file `stl_algobase.h`.

Referenced by `std::__move_merge()`, `__gnu_parallel::__parallel_random_shuffle_drs()`, `__gnu_parallel::__parallel__sort_qs_divide()`, `__gnu_profile::__report()`, `__gnu_parallel::__search_template()`, `__gnu_parallel::__sequential__random_shuffle()`, `std::deque<_Tp, _Alloc>::__M_reallocate_map()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare()`, `std::basic_string<char>::compare()`, `std::basic_string<_CharT, _Traits, _Alloc>::compare()`, `std::time_get<_CharT, _InIter>::do_date_order()`, `std::codecvt<_InternT, _ExternT, encoding_state>::do_out()`, `std::fill_n()`, `std::generate_canonical()`, `std::discard_block_engine<_RandomNumberEngine, __p, __r>::min()`, `std::shuffle_order_engine<_RandomNumberEngine, __k>::min()`, `std::minmax_element()`, `__gnu_parallel::multiseq_partition()`, `__gnu_parallel::multiseq_selection()`, `std::negative_binomial_distribution<_IntType>::operator()()`, `std::basic_stringbuf<_CharT, _Traits, _Alloc>::overflow()`, `std::basic_istream<_CharT, _Traits>::readsome()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::rfind()`, `std::basic_string<_CharT, _Traits, _Alloc>::rfind()`, `std::basic_filebuf<_CharT, _Traits>::underflow()`, `std::wbuffer_convert<_Codecvt, _Elem, _Tr>::underflow()`, `std::basic_streambuf<_CharT, _Traits>::xsgetn()`, `std::basic_filebuf<_CharT, _Traits>::xsputn()`, and `std::basic_streambuf<_CharT, _Traits>::xsputn()`.

3.71.2.16 `template<typename _Tp, typename _Compare> _GLIBCXX14_CONSTEXPR const _Tp & std::min (const _Tp & __a, const _Tp & __b, _Compare __comp) [inline]`

This does what you think it does.

Parameters

<code>__a</code>	A thing of arbitrary type.
<code>__b</code>	Another thing of arbitrary type.
<code>__comp</code>	A comparison functor .

Returns

The lesser of the parameters.

This will work on temporary expressions, since they are only evaluated once, unlike a preprocessor macro.

Definition at line 243 of file `stl_algobase.h`.

3.71.2.17 `template<typename _ForwardIterator> _GLIBCXX14_CONSTEXPR _ForwardIterator std::min_element (_ForwardIterator __first, _ForwardIterator __last) [inline]`

Return the minimum element in a range.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.

Returns

Iterator referencing the first instance of the smallest value.

Definition at line 5475 of file `stl_algo.h`.

```
3.71.2.18  template<typename _ForwardIterator, typename _Compare> _GLIBCXX14_CONSTEXPR _ForwardIterator
           std::min_element ( _ForwardIterator __first, _ForwardIterator __last, _Compare __comp ) [inline]
```

Return the minimum element in a range using comparison functor.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__comp</code>	Comparison functor.

Returns

Iterator referencing the first instance of the smallest value according to `__comp`.

Definition at line 5500 of file `stl_algo.h`.

Referenced by `std::minmax_element()`.

```
3.71.2.19  template<typename _Tp> _GLIBCXX14_CONSTEXPR pair< const _Tp &, const _Tp &> std::minmax ( const _Tp & __a,
           const _Tp & __b ) [inline]
```

Determines min and max at once as an ordered pair.

Parameters

<code>__a</code>	A thing of arbitrary type.
<code>__b</code>	Another thing of arbitrary type.

Returns

A pair(`__b`, `__a`) if `__b` is smaller than `__a`, pair(`__a`, `__b`) otherwise.

Definition at line 3299 of file `stl_algo.h`.

Referenced by `std::minmax_element()`.

```
3.71.2.20  template<typename _Tp, typename _Compare> _GLIBCXX14_CONSTEXPR pair< const _Tp &, const _Tp &>
           std::minmax ( const _Tp & __a, const _Tp & __b, _Compare __comp ) [inline]
```

Determines min and max at once as an ordered pair.

Parameters

<code>__a</code>	A thing of arbitrary type.
<code>__b</code>	Another thing of arbitrary type.
<code>__comp</code>	A comparison functor .

Returns

A pair(`__b`, `__a`) if `__b` is smaller than `__a`, pair(`__a`, `__b`) otherwise.

Definition at line 3320 of file `stl_algo.h`.

References `std::make_pair()`.

```
3.71.2.21  template<typename _ForwardIterator > _GLIBCXX14_CONSTEXPR pair<_ForwardIterator, _ForwardIterator>
            std::minmax_element ( _ForwardIterator __first, _ForwardIterator __last )  [inline]
```

Return a pair of iterators pointing to the minimum and maximum elements in a range.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.

Returns

`make_pair(m, M)`, where `m` is the first iterator `i` in `[__first, __last)` such that no other element in the range is smaller, and where `M` is the last iterator `i` in `[__first, __last)` such that no other element in the range is larger.

Definition at line 3400 of file `stl_algo.h`.

```
3.71.2.22  template<typename _ForwardIterator, typename _Compare > _GLIBCXX14_CONSTEXPR pair<_ForwardIterator,
            _ForwardIterator> std::minmax_element ( _ForwardIterator __first, _ForwardIterator __last, _Compare __comp )
            [inline]
```

Return a pair of iterators pointing to the minimum and maximum elements in a range.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__comp</code>	Comparison functor.

Returns

make_pair(m, M), where m is the first iterator i in [`__first`, `__last`) such that no other element in the range is smaller, and where M is the last iterator i in [`__first`, `__last`) such that no other element in the range is larger.

Definition at line 3428 of file `stl_algo.h`.

References `std::__find_if()`, `std::advance()`, `std::distance()`, `std::pair<_T1, _T2>::first`, `std::make_pair()`, `std::max()`, `std::max_element()`, `std::min()`, `std::min_element()`, `std::minmax()`, `std::minmax_element()`, and `std::pair<_T1, _T2>::second`.

Referenced by `std::minmax_element()`.

3.71.2.23 `template<typename _BidirectionalIterator> bool std::next_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last) [inline]`

Permute range into the next *dictionary* ordering.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.

Returns

False if wrapped to first permutation, true otherwise.

Treats all permutations of the range as a set of *dictionary* sorted sequences. Permutes the current sequence into the next one of this set. Returns true if there are more sequences to generate. If the sequence is the largest of the set, the smallest is generated and false returned.

Definition at line 2951 of file `stl_algo.h`.

3.71.2.24 `template<typename _BidirectionalIterator, typename _Compare> bool std::next_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare __comp) [inline]`

Permute range into the next *dictionary* ordering using comparison functor.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__comp</code>	A comparison functor.

Returns

False if wrapped to first permutation, true otherwise.

Treats all permutations of the range `[__first,__last)` as a set of *dictionary* sorted sequences ordered by `__comp`. Permutes the current sequence into the next one of this set. Returns true if there are more sequences to generate. If the sequence is the largest of the set, the smallest is generated and false returned.

Definition at line 2983 of file `stl_algo.h`.

References `std::__iterator_category()`, `std::__reverse()`, and `std::iter_swap()`.

3.71.2.25 `template<typename _RandomAccessIterator > void std::nth_element (_RandomAccessIterator __first, _RandomAccessIterator __nth, _RandomAccessIterator __last) [inline]`

Sort a sequence just enough to find a particular position.

Parameters

<code>__first</code>	An iterator.
<code>__nth</code>	Another iterator.
<code>__last</code>	Another iterator.

Returns

Nothing.

Rearranges the elements in the range `[__first,__last)` so that `*__nth` is the same element that would have been in that position had the whole sequence been sorted. The elements either side of `*__nth` are not completely sorted, but for any iterator *i* in the range `[__first,__nth)` and any iterator *j* in the range `[__nth,__last)` it holds that `*j < *i` is false.

Definition at line 4621 of file `stl_algo.h`.

References `std::lg()`.

3.71.2.26 `template<typename _RandomAccessIterator, typename _Compare > void std::nth_element (_RandomAccessIterator __first, _RandomAccessIterator __nth, _RandomAccessIterator __last, _Compare __comp) [inline]`

Sort a sequence just enough to find a particular position using a predicate for comparison.

Parameters

<code>__first</code>	An iterator.
<code>__nth</code>	Another iterator.
<code>__last</code>	Another iterator.
<code>__comp</code>	A comparison functor.

Returns

Nothing.

Rearranges the elements in the range `[__first,__last)` so that `*__nth` is the same element that would have been in that position had the whole sequence been sorted. The elements either side of `*__nth` are not completely sorted, but for

any iterator i in the range $[_\text{first}, _\text{nth})$ and any iterator j in the range $[_\text{nth}, _\text{last})$ it holds that `__comp(*j, *i)` is false.

Definition at line 4660 of file `stl_algo.h`.

References `std::__lg()`.

3.71.2.27 `template<typename _RandomAccessIterator > void std::partial_sort (_RandomAccessIterator __first, _RandomAccessIterator __middle, _RandomAccessIterator __last) [inline]`

Sort the smallest elements of a sequence.

Parameters

<code>__first</code>	An iterator.
<code>__middle</code>	Another iterator.
<code>__last</code>	Another iterator.

Returns

Nothing.

Sorts the smallest (`__middle-__first`) elements in the range $[\text{first}, \text{last})$ and moves them to the range $[_\text{first}, _\text{middle})$. The order of the remaining elements in the range $[_\text{middle}, _\text{last})$ is undefined. After the sort if i and j are iterators in the range $[_\text{first}, _\text{middle})$ such that i precedes j and k is an iterator in the range $[_\text{middle}, _\text{last})$ then $*j < *i$ and $*k < *i$ are both false.

Definition at line 4547 of file `stl_algo.h`.

3.71.2.28 `template<typename _RandomAccessIterator, typename _Compare > void std::partial_sort (_RandomAccessIterator __first, _RandomAccessIterator __middle, _RandomAccessIterator __last, _Compare __comp) [inline]`

Sort the smallest elements of a sequence using a predicate for comparison.

Parameters

<code>__first</code>	An iterator.
<code>__middle</code>	Another iterator.
<code>__last</code>	Another iterator.
<code>__comp</code>	A comparison functor.

Returns

Nothing.

Sorts the smallest (`__middle-__first`) elements in the range $[_\text{first}, _\text{last})$ and moves them to the range $[_\text{first}, _\text{middle})$. The order of the remaining elements in the range $[_\text{middle}, _\text{last})$ is undefined. After the sort if i and j are

iterators in the range `[__first,__middle)` such that `i` precedes `j` and `k` is an iterator in the range `[__middle,__last)` then `*__comp(j,*i)` and `__comp(*k,*i)` are both false.

Definition at line 4585 of file `stl_algo.h`.

```
3.71.2.29 template<typename _InputIterator, typename _RandomAccessIterator > _RandomAccessIterator std::partial_sort_copy (
    _InputIterator __first, _InputIterator __last, _RandomAccessIterator __result_first, _RandomAccessIterator __result_last )
    [inline]
```

Copy the smallest elements of a sequence.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__result_first</code>	A random-access iterator.
<code>__result_last</code>	Another random-access iterator.

Returns

An iterator indicating the end of the resulting sequence.

Copies and sorts the smallest `N` values from the range `[__first,__last)` to the range beginning at `__result_first`, where the number of elements to be copied, `N`, is the smaller of `(__last-__first)` and `(__result_last-__result_first)`. After the sort if `i` and `j` are iterators in the range `[__result_first,__result_first+N)` such that `i` precedes `j` then `*j<*i` is false. The value returned is `__result_first+N`.

Definition at line 1734 of file `stl_algo.h`.

```
3.71.2.30 template<typename _InputIterator, typename _RandomAccessIterator, typename _Compare > _RandomAccessIterator
    std::partial_sort_copy ( _InputIterator __first, _InputIterator __last, _RandomAccessIterator __result_first,
        _RandomAccessIterator __result_last, _Compare __comp ) [inline]
```

Copy the smallest elements of a sequence using a predicate for comparison.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	Another input iterator.
<code>__result_first</code>	A random-access iterator.
<code>__result_last</code>	Another random-access iterator.
<code>__comp</code>	A comparison functor.

Returns

An iterator indicating the end of the resulting sequence.

Copies and sorts the smallest `N` values from the range `[__first,__last)` to the range beginning at `result_first`, where the number of elements to be copied, `N`, is the smaller of `(__last-__first)` and `(__result_last-__result_first)`. After

the sort if i and j are iterators in the range $[_\text{result_first}, _\text{result_first} + N)$ such that i precedes j then $__\text{comp}(*j, *i)$ is false. The value returned is $__\text{result_first} + N$.

Definition at line 1784 of file `stl_algo.h`.

3.71.2.31 `template<typename _BidirectionalIterator > bool std::prev_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last) [inline]`

Permute range into the previous *dictionary* ordering.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.

Returns

False if wrapped to last permutation, true otherwise.

Treats all permutations of the range as a set of *dictionary* sorted sequences. Permutes the current sequence into the previous one of this set. Returns true if there are more sequences to generate. If the sequence is the smallest of the set, the largest is generated and false returned.

Definition at line 3051 of file `stl_algo.h`.

3.71.2.32 `template<typename _BidirectionalIterator, typename _Compare > bool std::prev_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare __comp) [inline]`

Permute range into the previous *dictionary* ordering using comparison functor.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__comp</code>	A comparison functor.

Returns

False if wrapped to last permutation, true otherwise.

Treats all permutations of the range $[_\text{first}, _\text{last})$ as a set of *dictionary* sorted sequences ordered by $__\text{comp}$. Permutes the current sequence into the previous one of this set. Returns true if there are more sequences to generate. If the sequence is the smallest of the set, the largest is generated and false returned.

Definition at line 3083 of file `stl_algo.h`.

3.71.2.33 `template<typename _RandomAccessIterator > void std::sort (_RandomAccessIterator __first, _RandomAccessIterator __last) [inline]`

Sort the elements of a sequence.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.

Returns

Nothing.

Sorts the elements in the range `[__first,__last)` in ascending order, such that for each iterator i in the range `[__first,__last-1)`, $*(i+1) < *i$ is false.

The relative ordering of equivalent elements is not preserved, use `stable_sort()` if this is needed.

Definition at line 4697 of file `stl_algo.h`.

```
3.71.2.34 template<typename _RandomAccessIterator, typename _Compare> void std::sort ( _RandomAccessIterator __first,
    _RandomAccessIterator __last, _Compare __comp ) [inline]
```

Sort the elements of a sequence using a predicate for comparison.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__comp</code>	A comparison functor.

Returns

Nothing.

Sorts the elements in the range `[__first,__last)` in ascending order, such that `__comp(*(i+1),*i)` is false for every iterator i in the range `[__first,__last-1)`.

The relative ordering of equivalent elements is not preserved, use `stable_sort()` if this is needed.

Definition at line 4727 of file `stl_algo.h`.

```
3.71.2.35 template<typename _RandomAccessIterator> void std::stable_sort ( _RandomAccessIterator __first,
    _RandomAccessIterator __last ) [inline]
```

Sort the elements of a sequence, preserving the relative order of equivalent elements.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.

Returns

Nothing.

Sorts the elements in the range $[_first, _last)$ in ascending order, such that for each iterator i in the range $[_first, _last-1)$, $*(i+1) < *i$ is false.

The relative ordering of equivalent elements is preserved, so any two elements x and y in the range $[_first, _last)$ such that $x < y$ is false and $y < x$ is false will have the same relative ordering after calling `stable_sort()`.

Definition at line 4902 of file `stl_algo.h`.

3.71.2.36 `template<typename _RandomAccessIterator, typename _Compare> void std::stable_sort (_RandomAccessIterator
__first, _RandomAccessIterator __last, _Compare __comp) [inline]`

Sort the elements of a sequence using a predicate for comparison, preserving the relative order of equivalent elements.

Parameters

<code>__first</code>	An iterator.
<code>__last</code>	Another iterator.
<code>__comp</code>	A comparison functor.

Returns

Nothing.

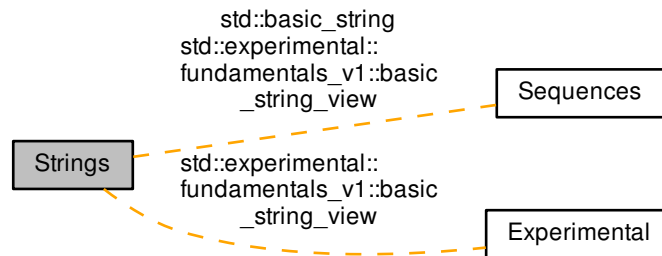
Sorts the elements in the range $[_first, _last)$ in ascending order, such that for each iterator i in the range $[_first, _last-1)$, `__comp(*(i+1),*i)` is false.

The relative ordering of equivalent elements is preserved, so any two elements x and y in the range $[_first, _last)$ such that `__comp(x, y)` is false and `__comp(y, x)` is false will have the same relative ordering after calling `stable_sort()`.

Definition at line 4936 of file `stl_algo.h`.

3.72 Strings

Collaboration diagram for Strings:



Classes

- class `std::basic_string< _CharT, _Traits, _Alloc >`
- struct `std::char_traits< _CharT >`
- class `std::experimental::fundamentals_v1::basic_string_view< _CharT, _Traits >`

Typedefs

- typedef `basic_string< char >` `std::string`
- typedef `basic_string< char16_t >` `std::u16string`
- typedef `basic_string< char32_t >` `std::u32string`
- typedef `basic_string< wchar_t >` `std::wstring`

3.72.1 Detailed Description

3.72.2 Typedef Documentation

3.72.2.1 typedef `basic_string<char>` `std::string`

A string of `char`.

Definition at line 71 of file `stringfwd.h`.

3.72.2.2 typedef `basic_string<char16_t>` `std::u16string`

A string of `char16_t`.

Definition at line 84 of file `stringfwd.h`.

3.72.2.3 `typedef basic_string<char32_t> std::u32string`

A string of `char32_t`.

Definition at line 87 of file `stringfwd.h`.

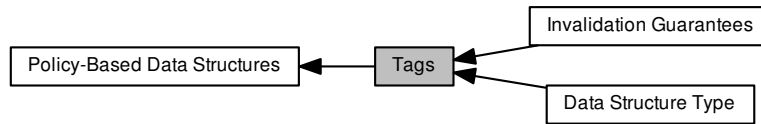
3.72.2.4 `typedef basic_string<wchar_t> std::wstring`

A string of `wchar_t`.

Definition at line 78 of file `stringfwd.h`.

3.73 Tags

Collaboration diagram for Tags:



Modules

- [Data Structure Type](#)
- [Invalidation Guarantees](#)

Classes

- [struct `__gnu_pbds::trivial_iterator_tag`](#)

Typedefs

- [typedef void `__gnu_pbds::trivial_iterator_difference_type`](#)

3.73.1 Detailed Description

3.73.2 Typedef Documentation

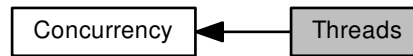
3.73.2.1 typedef void `__gnu_pbds::trivial_iterator_difference_type`

Prohibit moving trivial iterators.

Definition at line 79 of file `tag_and_trait.hpp`.

3.74 Threads

Collaboration diagram for Threads:



Namespaces

- [std::this_thread](#)

Classes

- struct [std::hash< thread::id >](#)
- class [std::thread](#)

Functions

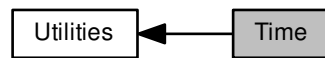
- bool **std::operator!=** (thread::id __x, thread::id __y) noexcept
- template<class _CharT, class _Traits >
basic_ostream< _CharT, _Traits > & **std::operator<<** (basic_ostream< _CharT, _Traits > &__out, thread::id __id)
- bool **std::operator<=** (thread::id __x, thread::id __y) noexcept
- bool **std::operator>** (thread::id __x, thread::id __y) noexcept
- bool **std::operator>=** (thread::id __x, thread::id __y) noexcept
- void **std::swap** (thread &__x, thread &__y) noexcept

3.74.1 Detailed Description

Classes for thread support.

3.75 Time

Collaboration diagram for Time:



Namespaces

- [std::chrono](#)

Macros

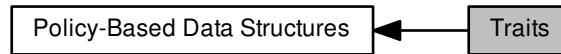
- `#define __cpp_lib_chrono_udls`

3.75.1 Detailed Description

Classes and functions for time.

3.76 Traits

Collaboration diagram for Traits:



Classes

- struct `__gnu_pbds::container_traits< Cntnr >`
- struct `__gnu_pbds::container_traits_base< _Tag >`
- struct `__gnu_pbds::container_traits_base< binary_heap_tag >`
- struct `__gnu_pbds::container_traits_base< binomial_heap_tag >`
- struct `__gnu_pbds::container_traits_base< cc_hash_tag >`
- struct `__gnu_pbds::container_traits_base< gp_hash_tag >`
- struct `__gnu_pbds::container_traits_base< list_update_tag >`
- struct `__gnu_pbds::container_traits_base< ov_tree_tag >`
- struct `__gnu_pbds::container_traits_base< pairing_heap_tag >`
- struct `__gnu_pbds::container_traits_base< pat_trie_tag >`
- struct `__gnu_pbds::container_traits_base< rb_tree_tag >`
- struct `__gnu_pbds::container_traits_base< rc_binomial_heap_tag >`
- struct `__gnu_pbds::container_traits_base< splay_tree_tag >`
- struct `__gnu_pbds::container_traits_base< thin_heap_tag >`
- struct `__gnu_pbds::detail::bin_search_tree_traits< Key, Mapped, Cmp_Fn, Node_Update, Node, _Alloc >`
- struct `__gnu_pbds::detail::bin_search_tree_traits< Key, null_type, Cmp_Fn, Node_Update, Node, _Alloc >`
- struct `__gnu_pbds::detail::no_throw_copies< Key, Mapped >`
- struct `__gnu_pbds::detail::no_throw_copies< Key, null_type >`
- struct `__gnu_pbds::detail::stored_data< _Tv, _Th >`
- struct `__gnu_pbds::detail::stored_data< _Tv, null_type >`
- struct `__gnu_pbds::detail::stored_hash< _Th >`
- struct `__gnu_pbds::detail::stored_value< _Tv >`
- struct `__gnu_pbds::detail::tree_metadata_helper< Node_Update, _BTp >`
- struct `__gnu_pbds::detail::tree_metadata_helper< Node_Update, false >`
- struct `__gnu_pbds::detail::tree_metadata_helper< Node_Update, true >`
- struct `__gnu_pbds::detail::tree_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc >`
- struct `__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >`
- struct `__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >`
- struct `__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >`
- struct `__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >`
- struct `__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >`
- struct `__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >`
- struct `__gnu_pbds::detail::trie_metadata_helper< Node_Update, _BTp >`
- struct `__gnu_pbds::detail::trie_metadata_helper< Node_Update, false >`

- struct `__gnu_pbds::detail::trie_metadata_helper< Node_Update, true >`
- struct `__gnu_pbds::detail::trie_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc >`
- struct `__gnu_pbds::detail::trie_traits< Key, Mapped, _ATraits, Node_Update, pat_trie_tag, _Alloc >`
- struct `__gnu_pbds::detail::trie_traits< Key, null_type, _ATraits, Node_Update, pat_trie_tag, _Alloc >`
- struct `__gnu_pbds::detail::type_base< Key, Mapped, _Alloc, Store_Hash >`
- struct `__gnu_pbds::detail::type_base< Key, Mapped, _Alloc, false >`
- struct `__gnu_pbds::detail::type_base< Key, Mapped, _Alloc, true >`
- struct `__gnu_pbds::detail::type_base< Key, null_type, _Alloc, false >`
- struct `__gnu_pbds::detail::type_base< Key, null_type, _Alloc, true >`
- struct `__gnu_pbds::detail::type_dispatch< Key, Mapped, _Alloc, Store_Hash >`
- struct `__gnu_pbds::detail::types_traits< Key, Mapped, _Alloc, Store_Hash >`
- struct `__gnu_pbds::null_node_update< _Tp1, _Tp2, _Tp3, _Tp4 >`
- struct `__gnu_pbds::null_type`

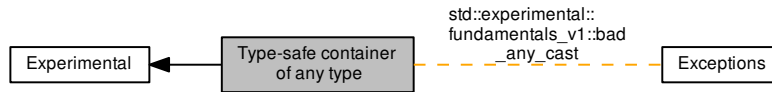
Variables

- static null_type `__gnu_pbds::detail::type_base< Key, null_type, _Alloc, false >::s_null_type`
- static null_type `__gnu_pbds::detail::type_base< Key, null_type, _Alloc, true >::s_null_type`

3.76.1 Detailed Description

3.77 Type-safe container of any type

Collaboration diagram for Type-safe container of any type:



Classes

- class `std::experimental::fundamentals_v1::any`
- class `std::experimental::fundamentals_v1::bad_any_cast`

Macros

- `#define __cpp_lib_experimental_any`

Functions

- `template<typename _Tp >`
`void * std::experimental::fundamentals_v1::__any_caster (const any *__any)`
- `void std::experimental::fundamentals_v1::__throw_bad_any_cast ()`
- `static void std::experimental::fundamentals_v1::any::__Manager_internal< _Tp >::_S_manage (_Op __↵`
`which, const any *__anyp, _Arg *__arg)`
- `static void std::experimental::fundamentals_v1::any::__Manager_external< _Tp >::_S_manage (_Op __↵`
`which, const any *__anyp, _Arg *__arg)`
- `template<typename _ValueType >`
`_ValueType std::experimental::fundamentals_v1::any_cast (const any &__any)`
- `void std::experimental::fundamentals_v1::swap (any &__x, any &__y) noexcept`
- `template<typename _ValueType >`
`_ValueType std::experimental::fundamentals_v1::any_cast (any &__any)`
- `template<typename _ValueType , typename enable_if<is_move_constructible< _ValueType >::value || is_lvalue_reference< _ValueType`
`>::value, bool >::type = true>`
`_ValueType std::experimental::fundamentals_v1::any_cast (any &&__any)`
- `template<typename _ValueType >`
`const _ValueType * std::experimental::fundamentals_v1::any_cast (const any *__any) noexcept`
- `template<typename _ValueType >`
`_ValueType * std::experimental::fundamentals_v1::any_cast (any *__any) noexcept`

3.77.1 Detailed Description

A type-safe container for single values of value types, as described in n3804 "Any Library Proposal (Revision 3)".

3.77.2 Function Documentation

3.77.2.1 `template<typename _ValueType > _ValueType std::experimental::fundamentals_v1::any_cast (const any & __any)`
`[inline]`

Access the contained object.

Template Parameters

<code>_ValueType</code>	A const-reference or CopyConstructible type.
-------------------------	--

Parameters

<code>__any</code>	The object to access.
--------------------	-----------------------

Returns

The contained object.

Exceptions

<code>bad_any_cast</code>	If <code>__any.type() != typeid(remove_reference_t<_ValueType>)</code>
---------------------------	--

Definition at line 363 of file any.

3.77.2.2 `template<typename _ValueType > _ValueType std::experimental::fundamentals_v1::any_cast (any & __any)`
`[inline]`

Access the contained object.

Template Parameters

<code>_ValueType</code>	A reference or CopyConstructible type.
-------------------------	--

Parameters

<code>__any</code>	The object to access.
--------------------	-----------------------

Returns

The contained object.

Exceptions

<i>bad_any_cast</i>	If <code>__any.type() != typeid(remove_reference_t<_ValueType>)</code>
---------------------	--

Definition at line 386 of file any.

```
3.77.2.3 template<typename _ValueType, typename enable_if<!is_move_constructible<_ValueType>::value || is_lvalue_
reference<_ValueType>::value, bool >::type = true> _ValueType std::experimental::fundamentals_v1::any_cast ( any
&& __any ) [inline]
```

Access the contained object.

Template Parameters

<i>_ValueType</i>	A reference or CopyConstructible type.
-------------------	--

Parameters

<i>__any</i>	The object to access.
--------------	-----------------------

Returns

The contained object.

Exceptions

<i>bad_any_cast</i>	If <code>__any.type() != typeid(remove_reference_t<_ValueType>)</code>
---------------------	--

Definition at line 400 of file any.

```
3.77.2.4 template<typename _ValueType> const _ValueType* std::experimental::fundamentals_v1::any_cast ( const any * __any
) [inline], [noexcept]
```

Access the contained object.

Template Parameters

<i>_ValueType</i>	The type of the contained object.
-------------------	-----------------------------------

Parameters

<code>__any</code>	A pointer to the object to access.
--------------------	------------------------------------

Returns

The address of the contained object if `__any != nullptr && __any.type() == typeid(_ValueType)` , otherwise a null pointer.

Definition at line 447 of file any.

```
3.77.2.5 template<typename _ValueType > _ValueType* std::experimental::fundamentals_v1::any_cast ( any * __any )
[inline],[noexcept]
```

Access the contained object.

Template Parameters

<code>_ValueType</code>	The type of the contained object.
-------------------------	-----------------------------------

Parameters

<code>__any</code>	A pointer to the object to access.
--------------------	------------------------------------

Returns

The address of the contained object if `__any != nullptr && __any.type() == typeid(_ValueType)` , otherwise a null pointer.

Definition at line 455 of file any.

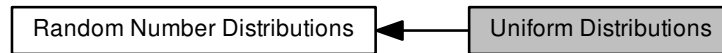
```
3.77.2.6 void std::experimental::fundamentals_v1::swap ( any & __x, any & __y ) [inline],[noexcept]
```

Exchange the states of two `any` objects.

Definition at line 350 of file any.

3.78 Uniform Distributions

Collaboration diagram for Uniform Distributions:



Classes

- class `std::uniform_real_distribution<_RealType>`

Functions

- `template<typename _IntType>`
`bool std::operator!=(const std::uniform_int_distribution<_IntType> &__d1, const std::uniform_int_distribution<_IntType> &__d2)`
- `template<typename _IntType>`
`bool std::operator!=(const std::uniform_real_distribution<_IntType> &__d1, const std::uniform_real_distribution<_IntType> &__d2)`
- `template<typename _IntType, typename _CharT, typename _Traits>`
`std::basic_ostream<_CharT, _Traits> & std::operator<< (std::basic_ostream<_CharT, _Traits> &, const std::uniform_int_distribution<_IntType> &)`
- `template<typename _RealType, typename _CharT, typename _Traits>`
`std::basic_ostream<_CharT, _Traits> & std::operator<< (std::basic_ostream<_CharT, _Traits> &, const std::uniform_real_distribution<_RealType> &)`
- `template<typename _IntType, typename _CharT, typename _Traits>`
`std::basic_istream<_CharT, _Traits> & std::operator>> (std::basic_istream<_CharT, _Traits> &, std::uniform_int_distribution<_IntType> &)`
- `template<typename _RealType, typename _CharT, typename _Traits>`
`std::basic_istream<_CharT, _Traits> & std::operator>> (std::basic_istream<_CharT, _Traits> &, std::uniform_real_distribution<_RealType> &)`

3.78.1 Detailed Description

3.78.2 Function Documentation

- 3.78.2.1 `template<typename _IntType> bool std::operator!=(const std::uniform_int_distribution<_IntType> & __d1, const std::uniform_int_distribution<_IntType> & __d2) [inline]`

Return true if two uniform integer distributions have different parameters.

Definition at line 1660 of file random.h.

References `std::__detail::operator>>()`.

3.78.2.2 `template<typename _IntType> bool std::operator!= (const std::uniform_real_distribution< _IntType> & __d1,
const std::uniform_real_distribution< _IntType> & __d2) [inline]`

Return true if two uniform real distributions have different parameters.

Definition at line 1869 of file random.h.

References `std::__detail::operator>>()`.

3.78.2.3 `template<typename _IntType, typename _CharT, typename _Traits> std::basic_ostream< _CharT, _Traits> &
std::operator<< (std::basic_ostream< _CharT, _Traits> & __os, const std::uniform_int_distribution<
_IntType> & __x)`

Inserts a `uniform_int_distribution` random number distribution `__x` into the output stream `os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>uniform_int_distribution</code> random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

Definition at line 877 of file bits/random.tcc.

References `std::ios_base::flags()`, `std::left()`, and `std::scientific()`.

3.78.2.4 `template<typename _RealType, typename _CharT, typename _Traits> std::basic_ostream< _CharT, _Traits> &
std::operator<< (std::basic_ostream< _CharT, _Traits> & __os, const std::uniform_real_distribution<
_RealType> & __x)`

Inserts a `uniform_real_distribution` random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>uniform_real_distribution</code> random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

Definition at line 936 of file bits/random.tcc.

References `std::ios_base::flags()`, `std::left()`, and `std::scientific()`.

3.78.2.5 `template<typename _IntType , typename _CharT , typename _Traits > std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > & __is, std::uniform_int_distribution< _IntType > & __x)`

Extracts a `uniform_int_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>uniform_int_distribution</code> random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

Definition at line 898 of file `bits/random.tcc`.

References `std::dec()`, `std::ios_base::flags()`, `std::uniform_int_distribution< _IntType >::param()`, and `std::skipws()`.

3.78.2.6 `template<typename _RealType , typename _CharT , typename _Traits > std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > & __is, std::uniform_real_distribution< _RealType > & __x)`

Extracts a `uniform_real_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>uniform_real_distribution</code> random number generator engine.

Returns

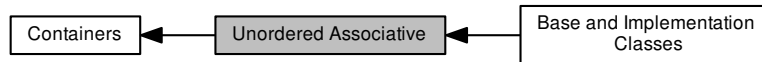
The input stream with `__x` extracted or in an error state.

Definition at line 960 of file `bits/random.tcc`.

References `std::ios_base::flags()`, `std::uniform_real_distribution< _RealType >::param()`, and `std::skipws()`.

3.79 Unordered Associative

Collaboration diagram for Unordered Associative:



Modules

- [Base and Implementation Classes](#)

Classes

- class `std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>`
- class `std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>`
- class `std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>`
- class `std::unordered_set<_Value, _Hash, _Pred, _Alloc>`

3.79.1 Detailed Description

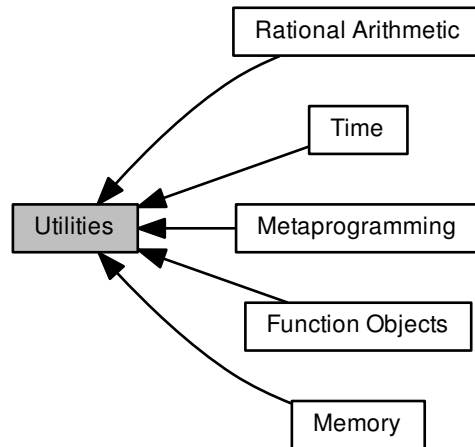
Unordered associative containers allow fast retrieval of data based on keys.

Each container type is parameterized on a `Key` type, a `Hash` type providing a hashing functor, and an ordering relation used to sort the elements of the container.

All unordered associative containers must meet certain requirements, summarized in [tables](#).

3.80 Utilities

Collaboration diagram for Utilities:



Modules

- [Function Objects](#)
- [Memory](#)
- [Metaprogramming](#)
- [Rational Arithmetic](#)
- [Time](#)

Classes

- `struct std::_Tuple_impl< _Idx, _Elements >`
- `struct std::_Tuple_impl< _Idx, _Head, _Tail... >`
- `class std::bitset< _Nb >`
- `struct std::pair< _T1, _T2 >`
- `struct std::piecewise_construct_t`
- `class std::tuple< _Elements >`
- `class std::tuple< _T1, _T2 >`
- `struct std::tuple_element< 0, tuple< _Head, _Tail... > >`
- `struct std::tuple_element< __i, tuple< _Head, _Tail... > >`
- `struct std::tuple_size< tuple< _Elements... > >`
- `struct std::type_index`
- `struct std::uses_allocator< tuple< _Types... >, _Alloc >`

Macros

- `#define __cpp_lib_tuples_by_type`

Typedefs

- `template<typename _Tp >`
`using std::__empty_not_final = typename conditional< __is_final(_Tp), false_type, __is_empty_non_tuple< _Tp >>::type`

Functions

- `template<typename... _Args1, typename... _Args2>`
`std::pair< _T1, _T2 >::pair (piecewise_construct_t, tuple< _Args1... >, tuple< _Args2... >)`
- `template<typename _Tp >`
`_Tp * std::__addressof (_Tp &__r) noexcept`
- `template<typename _Tp, typename _Up = _Tp>`
`_Tp std::__exchange (_Tp &__obj, _Up &&__new_val)`
- `template<std::size_t __i, typename _Head, typename... _Tail>`
`constexpr _Head & std::__get_helper (_Tuple_impl< __i, _Head, _Tail... > &__t) noexcept`
- `template<std::size_t __i, typename _Head, typename... _Tail>`
`constexpr const _Head & std::__get_helper (const _Tuple_impl< __i, _Head, _Tail... > &__t) noexcept`
- `template<typename _Head, size_t __i, typename... _Tail>`
`constexpr _Head & std::__get_helper2 (_Tuple_impl< __i, _Head, _Tail... > &__t) noexcept`
- `template<typename _Head, size_t __i, typename... _Tail>`
`constexpr const _Head & std::__get_helper2 (const _Tuple_impl< __i, _Head, _Tail... > &__t) noexcept`
- `template<typename _Tp >`
`_Tp * std::addressof (_Tp &__r) noexcept`
- `template<typename _Tp >`
`constexpr _Tp && std::forward (typename std::remove_reference< _Tp >::type &__t) noexcept`
- `template<typename _Tp >`
`constexpr _Tp && std::forward (typename std::remove_reference< _Tp >::type &&__t) noexcept`
- `template<typename... _Elements>`
`constexpr tuple< _Elements &&... > std::forward_as_tuple (_Elements &&... __args) noexcept`
- `template<std::size_t __i, typename... _Elements>`
`constexpr __tuple_element_t< __i, tuple< _Elements... > > & std::get (tuple< _Elements... > &__t) noexcept`
- `template<std::size_t __i, typename... _Elements>`
`constexpr const __tuple_element_t< __i, tuple< _Elements... > > & std::get (const tuple< _Elements... > &__t) noexcept`
- `template<std::size_t __i, typename... _Elements>`
`constexpr __tuple_element_t< __i, tuple< _Elements... > > && std::get (tuple< _Elements... > &&__t) noexcept`
- `template<typename _Tp, typename... _Types>`
`constexpr _Tp & std::get (tuple< _Types... > &__t) noexcept`
- `template<typename _Tp, typename... _Types>`
`constexpr _Tp && std::get (tuple< _Types... > &&__t) noexcept`
- `template<typename _Tp, typename... _Types>`
`constexpr const _Tp & std::get (const tuple< _Types... > &__t) noexcept`
- `template<typename _T1, typename _T2 >`
`constexpr pair< typename __decay_and_strip< _T1 >::type, typename __decay_and_strip< _T2 >::type > std::make_pair (_T1 &&__x, _T2 &&__y)`

- `template<typename... _Elements>`
`constexpr tuple< typename __decay_and_strip< _Elements >::__type... > std::make_tuple (_Elements`
`&&...__args)`
- `template<typename _Tp >`
`constexpr std::remove_reference< _Tp >::type && std::move (_Tp &&__t) noexcept`
- `template<typename _Tp >`
`constexpr conditional< __move_if_noexcept_cond< _Tp >::value, const _Tp &, _Tp && >::type std::move_if←`
`__noexcept (_Tp &__x) noexcept`
- `template<typename _T1 , typename _T2 >`
`constexpr bool std::operator!= (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool std::operator!= (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename _T1 , typename _T2 >`
`constexpr bool std::operator< (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool std::operator< (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename _T1 , typename _T2 >`
`constexpr bool std::operator<= (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool std::operator<= (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename _T1 , typename _T2 >`
`constexpr bool std::operator== (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool std::operator== (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename _T1 , typename _T2 >`
`constexpr bool std::operator> (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool std::operator> (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename _T1 , typename _T2 >`
`constexpr bool std::operator>= (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool std::operator>= (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename _Tp >`
`enable_if< __and< is_move_constructible< _Tp >, is_move_assignable< _Tp > >::value >::type std←`
`::swap (_Tp &__a, _Tp &__b) noexcept(__and< is_nothrow_move_constructible< _Tp >, is_nothrow_move←`
`_assignable< _Tp > >::value)`
- `template<typename _Tp , size_t _Nm>`
`enable_if< __is_swappable< _Tp >::value >::type std::swap (_Tp(&__a)[_Nm], _Tp(&__b)[_Nm]) noexcept(←`
`__is_nothrow_swappable< _Tp >::value)`
- `template<typename _T1 , typename _T2 >`
`void std::swap (pair< _T1, _T2 > &__x, pair< _T1, _T2 > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename... _Elements>`
`void std::swap (tuple< _Elements... > &__x, tuple< _Elements... > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename... _Elements>`
`constexpr tuple< _Elements &... > std::tie (_Elements &...__args) noexcept`
- `template<typename... _Tpls, typename = typename enable_if<__and<__is_tuple_like<_Tpls>...>::value>::type>`
`constexpr auto std::tuple_cat (_Tpls &&... __tpls) -> typename __tuple_cat_result< _Tpls... >::__type`

Variables

- `const _Swallow_assign std::ignore`
- `constexpr piecewise_construct_t std::piecewise_construct`

3.80.1 Detailed Description

Components deemed generally useful. Includes pair, tuple, forward/move helpers, ratio, function object, metaprogramming and type traits, time, date, and memory functions.

3.80.2 Function Documentation

3.80.2.1 `template<typename _Tp> _Tp* std::__addressof (_Tp &__r) [inline], [noexcept]`

Same as C++11 `std::addressof`.

Definition at line 47 of file `move.h`.

Referenced by `__gnu_debug::__check_dereferenceable()`, `std::_Destroy()`, `std::vector< sub_match< _Bi_iter >, allocator< sub_match< _Bi_iter > > >::_M_allocate_and_copy()`, `__gnu_cxx::bitmap_allocator< _Tp >::_M_deallocate_single_object()`, `std::addressof()`, `std::vector< _Tp, _Alloc >::emplace()`, `std::Temporary_buffer< _ForwardIterator, _Tp >::end()`, `std::list< _Tp, _Alloc >::merge()`, `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator->()`, `__gnu_debug::_Safe_iterator< _Iterator, _Sequence >::operator->()`, `std::forward_list< _Tp, _Alloc >::operator=()`, `std::list< _Tp, _Alloc >::operator=()`, `std::forward_list< _Tp, _Alloc >::remove()`, `std::list< _Tp, _Alloc >::remove()`, `std::rethrow_if_nested()`, `std::list< __inp, __rebind_inp >::splice()`, `std::uninitialized_copy()`, `std::uninitialized_fill()`, and `std::uninitialized_fill_n()`.

3.80.2.2 `template<typename _Tp> _Tp* std::addressof (_Tp &__r) [inline], [noexcept]`

Returns the actual address of the object or function referenced by `r`, even in the presence of an overloaded operator`&`.

Parameters

<code>↔</code>	Reference to an object or function.
<code>__↔</code>	
<code>↔</code>	
<code>__↔</code>	
<code>r</code>	

Returns

The actual address.

Definition at line 135 of file `move.h`.

References `std::__addressof()`.

Referenced by `__gnu_cxx::operator==()`, and `std::pointer_traits< _Tp * >::pointer_to()`.

3.80.2.3 `template<typename _Tp> constexpr _Tp&& std::forward (typename std::remove_reference< _Tp >::type &__t) [noexcept]`

Forward an lvalue.

Returns

The parameter cast to the specified type.

This function is used to implement "perfect forwarding".

Definition at line 76 of file move.h.

3.80.2.4 `template<typename _Tp> constexpr _Tp&& std::forward (typename std::remove_reference< _Tp >::type && __t)
[noexcept]`

Forward an rvalue.

Returns

The parameter cast to the specified type.

This function is used to implement "perfect forwarding".

Definition at line 87 of file move.h.

3.80.2.5 `template<std::size_t __i, typename... _Elements> constexpr __tuple_element_t<__i, tuple<_Elements...> >& std::get (tuple<_Elements...> & __t) [noexcept]`

Return a reference to the ith element of a tuple.

Definition at line 1254 of file tuple.

3.80.2.6 `template<std::size_t __i, typename... _Elements> constexpr const __tuple_element_t<__i, tuple<_Elements...> >& std::get (const tuple<_Elements...> & __t) [noexcept]`

Return a const reference to the ith element of a const tuple.

Definition at line 1260 of file tuple.

3.80.2.7 `template<std::size_t __i, typename... _Elements> constexpr __tuple_element_t<__i, tuple<_Elements...> >&& std::get (tuple<_Elements...> && __t) [noexcept]`

Return an rvalue reference to the ith element of a tuple rvalue.

Definition at line 1266 of file tuple.

3.80.2.8 `template<typename _Tp, typename... _Types> constexpr _Tp& std::get (tuple<_Types...> & __t) [noexcept]`

Return a reference to the unique element of type _Tp of a tuple.

Definition at line 1289 of file tuple.

3.80.2.9 `template<typename _Tp, typename... _Types> constexpr _Tp&& std::get (tuple< _Types... > && __t)`
[noexcept]

Return a reference to the unique element of type `_Tp` of a tuple rvalue.

Definition at line 1295 of file tuple.

3.80.2.10 `template<typename _Tp, typename... _Types> constexpr const _Tp& std::get (const tuple< _Types... > & __t)`
[noexcept]

Return a const reference to the unique element of type `_Tp` of a tuple.

Definition at line 1301 of file tuple.

3.80.2.11 `template<typename _T1, typename _T2 > constexpr pair<typename __decay_and_strip<_T1>::__type, typename __decay_and_strip<_T2>::__type> std::make_pair (_T1 && __x, _T2 && __y)`

A convenience wrapper for creating a pair from two objects.

Parameters

<code>_↵ _x</code>	The first object.
<code>_↵ _y</code>	The second object.

Returns

A newly-constructed `pair<>` object of the appropriate type.

The standard requires that the objects be passed by reference-to-const, but LWG issue #181 says they should be passed by const value. We follow the LWG by default.

Definition at line 493 of file `stl_pair.h`.

Referenced by `__gnu_parallel::__find_template()`, `__gnu_debug::__get_distance()`, `__gnu_parallel::__parallel_merge↵
_advance()`, `__gnu_parallel::__parallel_sort_qsb()`, `__gnu_parallel::__qsb_local_sort_with_helping()`, `__gnu_debug↵
::__valid_range_aux()`, `__gnu_parallel::__find_if_selector::__M_sequential_algorithm()`, `__gnu_parallel::__adjacent↵
_find_selector::__M_sequential_algorithm()`, `__gnu_parallel::__find_first_of_selector<_FIterator>::__M_sequential↵
_algorithm()`, `__gnu_pbds::detail::pat_trie_base::__Node_iter<Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc↵
>::get_child()`, `std::minmax()`, `std::minmax_element()`, `__gnu_parallel::multiseq_partition()`, `__gnu_parallel::multiseq↵
_selection()`, `__gnu_pbds::detail::pat_trie_map<Key, Mapped, Node_And_It_Traits, _Alloc>::node_end()`, `__gnu↵
_parallel::parallel_multiway_merge()`, `__gnu_parallel::parallel_sort_mwms_pu()`, and `__gnu_pbds::detail::pat_trie↵
base::__Node_citer<Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc>::valid_prefix()`.

3.80.2.12 `template<typename _Tp> constexpr std::remove_reference<_Tp>::type&& std::move (_Tp && __t)
[noexcept]`

Convert a value to an rvalue.

Parameters

<code>↵ _↵ ↵ _↵ t</code>	A thing of arbitrary type.
--	----------------------------

Returns

The parameter cast to an rvalue-reference to allow moving it.

Definition at line 101 of file `move.h`.

3.80.2.13 `template<typename _Tp> constexpr conditional<__move_if_noexcept_cond<_Tp>::value, const _Tp&, _Tp&&>::type
std::move_if_noexcept (_Tp & __x) [noexcept]`

Conditionally convert a value to an rvalue.

Parameters

<code>_↔</code>	A thing of arbitrary type.
<code>__x</code>	

Returns

The parameter, possibly cast to an rvalue-reference.

Same as `std::move` unless the type's move constructor could throw and the type is copyable, in which case an lvalue-reference is returned instead.

Definition at line 121 of file `move.h`.

```
3.80.2.14  template<typename _T1, typename _T2 > constexpr bool std::operator!= ( const pair< _T1, _T2 > &__x, const pair<
        _T1, _T2 > &__y )  [inline]
```

Uses `operator==` to find the result.

Definition at line 444 of file `stl_pair.h`.

```
3.80.2.15  template<typename _T1, typename _T2 > constexpr bool std::operator< ( const pair< _T1, _T2 > &__x, const pair<
        _T1, _T2 > &__y )  [inline]
```

<http://gcc.gnu.org/onlinedocs/libstdc++/manual/utilities.html>

Definition at line 437 of file `stl_pair.h`.

References `std::pair< _T1, _T2 >::first`.

```
3.80.2.16  template<typename _T1, typename _T2 > constexpr bool std::operator<= ( const pair< _T1, _T2 > &__x, const
        pair< _T1, _T2 > &__y )  [inline]
```

Uses `operator<` to find the result.

Definition at line 456 of file `stl_pair.h`.

```
3.80.2.17  template<typename _T1, typename _T2 > constexpr bool std::operator== ( const pair< _T1, _T2 > &__x, const
        pair< _T1, _T2 > &__y )  [inline]
```

Two pairs of the same type are equal iff their members are equal.

Definition at line 431 of file `stl_pair.h`.

References `std::pair< _T1, _T2 >::first`, and `std::pair< _T1, _T2 >::second`.

3.80.2.18 `template<typename _T1, typename _T2> constexpr bool std::operator> (const pair<_T1, _T2> &__x, const pair<_T1, _T2> &__y) [inline]`

Uses `operator<` to find the result.

Definition at line 450 of file `stl_pair.h`.

3.80.2.19 `template<typename _T1, typename _T2> constexpr bool std::operator>= (const pair<_T1, _T2> &__x, const pair<_T1, _T2> &__y) [inline]`

Uses `operator<` to find the result.

Definition at line 462 of file `stl_pair.h`.

3.80.2.20 `template<typename _Tp> enable_if<__and<is_move_constructible<_Tp>, is_move_assignable<_Tp>>::value>::type std::swap (_Tp &__a, _Tp &__b) [inline], [noexcept]`

Swaps two values.

Parameters

\leftrightarrow _a	A thing of arbitrary type.
\leftrightarrow _b	Another thing of arbitrary type.

Returns

Nothing.

Definition at line 179 of file move.h.

```
3.80.2.21  template<typename _Tp, size_t _Nm> enable_if<__is_swappable<_Tp>::value>::type std::swap ( _Tp(&) __a[_Nm],
        _Tp(&) __b[_Nm] )  [inline], [noexcept]
```

Swap the contents of two arrays.

Definition at line 202 of file move.h.

```
3.80.2.22  template<typename _T1, typename _T2 > void std::swap ( pair<_T1, _T2 > &__x, pair<_T1, _T2 > &__y )
        [inline], [noexcept]
```

See std::pair::swap().

Definition at line 471 of file stl_pair.h.

```
3.80.2.23  template<typename... _Elements> void std::swap ( tuple<_Elements... > &__x, tuple<_Elements... > &__y )
        [inline], [noexcept]
```

swap

Definition at line 1546 of file tuple.

```
3.80.2.24  template<typename... _Elements> constexpr tuple<_Elements&...> std::tie ( _Elements &... __args )  [noexcept]
```

tie

Definition at line 1540 of file tuple.

Referenced by std::basic_ios<_CharT, _Traits >::copyfmt().

```
3.80.2.25  template<typename... _Tpls, typename = typename enable_if<__and<__is_tuple_like<_Tpls>...>::value>::type>
        constexpr auto std::tuple_cat ( _Tpls &&... __tpls )-> typename __tuple_cat_result<_Tpls...>::__type
```

tuple_cat

Definition at line 1526 of file tuple.

3.80.3 Variable Documentation

```
3.80.3.1  constexpr piecewise_construct_t std::piecewise_construct
```

piecewise_construct

Definition at line 79 of file stl_pair.h.

Referenced by std::unordered_map<_Key, _Tp, _Hash, _Pred >::emplace_hint(), std::map<_Key, _Tp, _Compare, _Alloc >::emplace_hint(), std::unordered_map<_Key, _Tp, _Hash, _Pred >::insert(), std::map<_Key, _Tp, _Compare, _Alloc >::insert(), and std::map<_Key, _Tp, _Compare, _Alloc >::operator[]().

4 Namespace Documentation

4.1 `__gnu_cxx` Namespace Reference

Namespaces

- [__detail](#)
- [typelist](#)

Classes

- [struct __alloc_traits](#)
- [struct __common_pool_policy](#)
- [class __mt_alloc](#)
- [class __mt_alloc_base](#)
- [struct __per_type_pool_policy](#)
- [class __pool](#)
- [class __pool< false >](#)
- [class __pool< true >](#)
- [class __pool_alloc](#)
- [class __pool_alloc_base](#)
- [struct __pool_base](#)
- [class __rc_string_base](#)
- [class __scoped_lock](#)
- [class __versa_string](#)
- [struct _Caster](#)
- [struct _Char_types](#)
- [class _ExtPtr_allocator](#)
- [struct _Invalid_type](#)
- [class _Pointer_adapter](#)
- [class _Relative_pointer_impl](#)
- [class _Relative_pointer_impl< const _Tp >](#)
- [class _Std_pointer_impl](#)
- [struct _Unqualified_type](#)
- [struct annotate_base](#)
- [class array_allocator](#)
- [class array_allocator_base](#)
- [class binary_compose](#)
- [class bitmap_allocator](#)
- [struct char_traits](#)
- [struct character](#)
- [struct condition_base](#)
- [struct constant_binary_fun](#)
- [struct constant_unary_fun](#)
- [struct constant_void_fun](#)
- [class debug_allocator](#)
- [class enc_filebuf](#)
- [struct encoding_char_traits](#)
- [class encoding_state](#)

- struct [forced_error](#)
- class [free_list](#)
- class [hash_map](#)
- class [hash_multimap](#)
- class [hash_multiset](#)
- class [hash_set](#)
- struct [limit_condition](#)
- class [malloc_allocator](#)
- class [new_allocator](#)
- struct [project1st](#)
- struct [project2nd](#)
- struct [random_condition](#)
- struct [rb_tree](#)
- class [recursive_init_error](#)
- class [rope](#)
- struct [select1st](#)
- struct [select2nd](#)
- class [slist](#)
- class [stdio_filebuf](#)
- class [stdio_sync_filebuf](#)
- class [subtractive_rng](#)
- struct [temporary_buffer](#)
- class [throw_allocator_base](#)
- struct [throw_allocator_limit](#)
- struct [throw_allocator_random](#)
- struct [throw_value_base](#)
- struct [throw_value_limit](#)
- struct [throw_value_random](#)
- class [unary_compose](#)

Typedefs

- typedef void(* **__destroy_handler**) (void *)
- typedef [__versa_string](#)< char, [std::char_traits](#)< char >, [std::allocator](#)< char >, [__rc_string_base](#) > **__rc_string**
- typedef [__vstring](#) **__sso_string**
- typedef [__versa_string](#)< char16_t, [std::char_traits](#)< char16_t >, [std::allocator](#)< char16_t >, [__rc_string_base](#) > **__u16rc_string**
- typedef [__u16vstring](#) **__u16sso_string**
- typedef [__versa_string](#)< char16_t > **__u16vstring**
- typedef [__versa_string](#)< char32_t, [std::char_traits](#)< char32_t >, [std::allocator](#)< char32_t >, [__rc_string_base](#) > **__u32rc_string**
- typedef [__u32vstring](#) **__u32sso_string**
- typedef [__versa_string](#)< char32_t > **__u32vstring**
- typedef [__versa_string](#)< char > **__vstring**
- typedef [__versa_string](#)< wchar_t, [std::char_traits](#)< wchar_t >, [std::allocator](#)< wchar_t >, [__rc_string_base](#) > **__wrc_string**
- typedef [__wvstring](#) **__wsso_string**
- typedef [__versa_string](#)< wchar_t > **__wvstring**
- typedef [rope](#)< char > **crope**
- typedef [rope](#)< wchar_t > **wrope**

Enumerations

- enum { **_S_num_primes** }
- enum **_Lock_policy** { **_S_single**, **_S_mutex**, **_S_atomic** }

Functions

- static void **__atomic_add_single** (**_Atomic_word** *__mem, int __val)
- else **__atomic_add_single** (__mem, __val)
- **_Atomic_word** **__attribute** ((__unused__)) **__exchange_and_add**(volatile **_Atomic_word** *
- namespace **__cxx11** **__attribute** ((__abi_tag__("cxx11")))
- template<class **_Tp** >
void **__aux_require_boolean_expr** (const **_Tp** &__t)
- template<typename **_ToType** , typename **_FromType** >
_ToType **__const_pointer_cast** (const **_FromType** &__arg)
- template<typename **_ToType** , typename **_FromType** >
_ToType **__const_pointer_cast** (**_FromType** *__arg)
- template<typename **_InputIterator** , typename **_Size** , typename **_OutputIterator** >
[pair](#)< **_InputIterator** , **_OutputIterator** > **__copy_n** (**_InputIterator** __first, **_Size** __count, **_OutputIterator** __result, [input_iterator_tag](#))
- template<typename **_RAIterator** , typename **_Size** , typename **_OutputIterator** >
[pair](#)< **_RAIterator** , **_OutputIterator** > **__copy_n** (**_RAIterator** __first, **_Size** __count, **_OutputIterator** __result, [random_access_iterator_tag](#))
- template<typename **_InputIterator** , typename **_Distance** >
void **__distance** (**_InputIterator** __first, **_InputIterator** __last, **_Distance** &__n, [std::input_iterator_tag](#))
- template<typename **_RandomAccessIterator** , typename **_Distance** >
void **__distance** (**_RandomAccessIterator** __first, **_RandomAccessIterator** __last, **_Distance** &__n, [std::random_access_iterator_tag](#))
- template<typename **_ToType** , typename **_FromType** >
_ToType **__dynamic_pointer_cast** (const **_FromType** &__arg)
- template<typename **_ToType** , typename **_FromType** >
_ToType **__dynamic_pointer_cast** (**_FromType** *__arg)
- void **__error_type_must_be_a_signed_integer_type** ()
- void **__error_type_must_be_an_integer_type** ()
- void **__error_type_must_be_an_unsigned_integer_type** ()
- static **_Atomic_word** **__exchange_and_add_single** (**_Atomic_word** *__mem, int __val)
- else return **__exchange_and_add_single** (__mem, __val)
- template<class **_Concept** >
void **__function_requires** ()
- template<typename **_Type** >
bool **__is_null_pointer** (**_Type** *__ptr)
- template<typename **_Type** >
bool **__is_null_pointer** (**_Type**)
- bool **__is_null_pointer** (std::nullptr_t)
- template<typename **_InputIterator1** , typename **_InputIterator2** >
int **__lexicographical_compare_3way** (**_InputIterator1** __first1, **_InputIterator1** __last1, **_InputIterator2** __first2, **_InputIterator2** __last2)
- int **__lexicographical_compare_3way** (const unsigned char *__first1, const unsigned char *__last1, const unsigned char *__first2, const unsigned char *__last2)
- int **__lexicographical_compare_3way** (const char *__first1, const char *__last1, const char *__first2, const char *__last2)

- `template<typename _Tp >`
`const _Tp & __median (const _Tp &__a, const _Tp &__b, const _Tp &__c)`
- `template<typename _Tp, typename _Compare >`
`const _Tp & __median (const _Tp &__a, const _Tp &__b, const _Tp &__c, _Compare __comp)`
- `crope::reference __mutable_reference_at (crope &__c, size_t __i)`
- `template<typename _Tp, typename _Integer, typename _MonoidOperation >`
`_Tp __power (_Tp __x, _Integer __n, _MonoidOperation __monoid_op)`
- `template<typename _Tp, typename _Integer >`
`_Tp __power (_Tp __x, _Integer __n)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _Distance >`
`_RandomAccessIterator __random_sample (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out, const _Distance __n)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _RandomNumberGenerator, typename _Distance >`
`_RandomAccessIterator __random_sample (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out, _RandomNumberGenerator &__rand, const _Distance __n)`
- `template<typename _ToType, typename _FromType >`
`_ToType __reinterpret_pointer_cast (const _FromType &__arg)`
- `template<typename _ToType, typename _FromType >`
`_ToType __reinterpret_pointer_cast (_FromType * __arg)`
- `_Slist_node_base * __slist_make_link (_Slist_node_base * __prev_node, _Slist_node_base * __new_node)`
- `_Slist_node_base * __slist_previous (_Slist_node_base * __head, const _Slist_node_base * __node)`
- `const _Slist_node_base * __slist_previous (const _Slist_node_base * __head, const _Slist_node_base * __↵ node)`
- `_Slist_node_base * __slist_reverse (_Slist_node_base * __node)`
- `size_t __slist_size (_Slist_node_base * __node)`
- `void __slist_splice_after (_Slist_node_base * __pos, _Slist_node_base * __before_first, _Slist_node_base * __↵ __before_last)`
- `void __slist_splice_after (_Slist_node_base * __pos, _Slist_node_base * __head)`
- `template<typename _ToType, typename _FromType >`
`_ToType __static_pointer_cast (const _FromType &__arg)`
- `template<typename _ToType, typename _FromType >`
`_ToType __static_pointer_cast (_FromType * __arg)`
- `size_t __stl_hash_string (const char * __s)`
- `unsigned long __stl_next_prime (unsigned long __n)`
- `template<typename _TRet, typename _Ret = _TRet, typename _CharT, typename... _Base>`
`_Ret __stoa (_TRet(* __convf)(const _CharT *, _CharT **, _Base...), const char * __name, const _CharT * __str, std::size_t * __idx, _Base... __base)`
- `void __throw_concurrency_lock_error ()`
- `void __throw_concurrency_unlock_error ()`
- `void __throw_forced_error ()`
- `template<typename _String, typename _CharT = typename _String::value_type>`
`_String __to_xstring (int(* __convf)(_CharT *, std::size_t, const _CharT *, __builtin_va_list), std::size_t __n, const _CharT * __fmt,...)`
- `template<typename _InputIter, typename _Size, typename _ForwardIter >`
`pair< _InputIter, _ForwardIter > __uninitialized_copy_n (_InputIter __first, _Size __count, _ForwardIter __↵ result, std::input_iterator_tag)`
- `template<typename _RandomAccessIter, typename _Size, typename _ForwardIter >`
`pair< _RandomAccessIter, _ForwardIter > __uninitialized_copy_n (_RandomAccessIter __first, _Size __count, _ForwardIter __result, std::random_access_iterator_tag)`
- `template<typename _InputIter, typename _Size, typename _ForwardIter >`
`pair< _InputIter, _ForwardIter > __uninitialized_copy_n (_InputIter __first, _Size __count, _ForwardIter __↵ result)`

- `template<typename _InputIter, typename _Size, typename _ForwardIter, typename _Allocator >`
`pair< _InputIter, _ForwardIter > uninitialized_copy_n_a (_InputIter __first, _Size __count, _ForwardIter __↵`
`__result, _Allocator __alloc)`
- `template<typename _InputIter, typename _Size, typename _ForwardIter, typename _Tp >`
`pair< _InputIter, _ForwardIter > uninitialized_copy_n_a (_InputIter __first, _Size __count, _ForwardIter __↵`
`__result, std::allocator< _Tp >)`
- `void __verbose_terminate_handler ()`
- `size_t _Bit_scan_forward (size_t __num)`
- `template<typename _ForwardIterator, typename _Allocator >`
`void _Destroy_const (_ForwardIterator __first, _ForwardIterator __last, _Allocator __alloc)`
- `template<typename _ForwardIterator, typename _Tp >`
`void _Destroy_const (_ForwardIterator __first, _ForwardIterator __last, allocator< _Tp >)`
- `template<class _CharT, class _Traits >`
`void _Rope_fill (basic_ostream< _CharT, _Traits > &__o, size_t __n)`
- `template<class _CharT >`
`bool _Rope_is_simple (_CharT *)`
- `bool _Rope_is_simple (char *)`
- `bool _Rope_is_simple (wchar_t *)`
- `template<class _Rope_iterator >`
`void _Rope_rotate (_Rope_iterator __first, _Rope_iterator __middle, _Rope_iterator __last)`
- `template<class _CharT >`
`void _S_cond_store_eos (_CharT &)`
- `void _S_cond_store_eos (char &__c)`
- `void _S_cond_store_eos (wchar_t &__c)`
- `template<class _CharT >`
`_CharT _S_eos (_CharT *)`
- `template<class _CharT >`
`bool _S_is_basic_char_type (_CharT *)`
- `bool _S_is_basic_char_type (char *)`
- `bool _S_is_basic_char_type (wchar_t *)`
- `template<class _CharT >`
`bool _S_is_one_byte_char_type (_CharT *)`
- `bool _S_is_one_byte_char_type (char *)`
- `template<class _Operation1, class _Operation2 >`
`unary_compose< _Operation1, _Operation2 > compose1 (const _Operation1 &__fn1, const _Operation2 &__↵`
`__fn2)`
- `template<class _Operation1, class _Operation2, class _Operation3 >`
`binary_compose< _Operation1, _Operation2, _Operation3 > compose2 (const _Operation1 &__fn1, const __↵`
`_Operation2 &__fn2, const _Operation3 &__fn3)`
- `template<typename _Tpa, typename _Tpc, typename _Tp >`
`__gnu_cxx::__promote_3< _Tpa, _Tpc, _Tp >::__type conf_hyperg (_Tpa __a, _Tpc __c, _Tp __x)`
- `float conf_hypergf (float __a, float __c, float __x)`
- `long double conf_hypergl (long double __a, long double __c, long double __x)`
- `template<class _Result >`
`constant_void_fun< _Result > constant0 (const _Result &__val)`
- `template<class _Result >`
`constant_unary_fun< _Result, _Result > constant1 (const _Result &__val)`
- `template<class _Result >`
`constant_binary_fun< _Result, _Result, _Result > constant2 (const _Result &__val)`
- `template<typename _InputIterator, typename _Size, typename _OutputIterator >`
`pair< _InputIterator, _OutputIterator > copy_n (_InputIterator __first, _Size __count, _OutputIterator __result)`

- `template<typename _InputIterator, typename _Tp, typename _Size >`
`void count (_InputIterator __first, _InputIterator __last, const _Tp &__value, _Size &__n)`
- `template<typename _InputIterator, typename _Predicate, typename _Size >`
`void count_if (_InputIterator __first, _InputIterator __last, _Predicate __pred, _Size &__n)`
- `template<typename _InputIterator, typename _Distance >`
`void distance (_InputIterator __first, _InputIterator __last, _Distance &__n)`
- `template<typename _Tpa, typename _Tpb, typename _Tpc, typename _Tp >`
`__gnu_cxx::__promote_4< _Tpa, _Tpb, _Tpc, _Tp >::__type hyperg (_Tpa __a, _Tpb __b, _Tpc __c, _Tp __x)`
- `float hypergf (float __a, float __b, float __c, float __x)`
- `long double hypergl (long double __a, long double __b, long double __c, long double __x)`
- `template<class _Tp >`
`_Tp identity_element (std::plus< _Tp >)`
- `template<class _Tp >`
`_Tp identity_element (std::multiplies< _Tp >)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`int lexicographical_compare_3way (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, ↵
_InputIterator2 __last2)`
- `template<class _Ret, class _Tp, class _Arg >`
`mem_fun1_t< _Ret, _Tp, _Arg > mem_fun1 (_Ret(_Tp::*__f)(_Arg))`
- `template<class _Ret, class _Tp, class _Arg >`
`const_mem_fun1_t< _Ret, _Tp, _Arg > mem_fun1 (_Ret(_Tp::*__f)(_Arg) const)`
- `template<class _Ret, class _Tp, class _Arg >`
`mem_fun1_ref_t< _Ret, _Tp, _Arg > mem_fun1_ref (_Ret(_Tp::*__f)(_Arg))`
- `template<class _Ret, class _Tp, class _Arg >`
`const_mem_fun1_ref_t< _Ret, _Tp, _Arg > mem_fun1_ref (_Ret(_Tp::*__f)(_Arg) const)`
- `template<typename _Tp >`
`bool operator!= (const new_allocator< _Tp > &, const new_allocator< _Tp > &)`
- `template<typename _Tp >`
`bool operator!= (const malloc_allocator< _Tp > &, const malloc_allocator< _Tp > &)`
- `template<typename _Tp, typename _Array >`
`bool operator!= (const array_allocator< _Tp, _Array > &, const array_allocator< _Tp, _Array > &)`
- `template<typename _Alloc >`
`bool operator!= (const debug_allocator< _Alloc > &__lhs, const debug_allocator< _Alloc > &__rhs)`
- `template<typename _Tp >`
`bool operator!= (const __pool_alloc< _Tp > &, const __pool_alloc< _Tp > &)`
- `template<class _Value, class _HashFcn, class _EqualKey, class _Alloc >`
`bool operator!= (const hash_set< _Value, _HashFcn, _EqualKey, _Alloc > &__hs1, const hash_set< _Value, ↵
_HashFcn, _EqualKey, _Alloc > &__hs2)`
- `template<class _Key, class _Tp, class _HashFn, class _EqKey, class _Alloc >`
`bool operator!= (const hash_map< _Key, _Tp, _HashFn, _EqKey, _Alloc > &__hm1, const hash_map< _Key, ↵
_Tp, _HashFn, _EqKey, _Alloc > &__hm2)`
- `template<class _Val, class _HashFcn, class _EqualKey, class _Alloc >`
`bool operator!= (const hash_multiset< _Val, _HashFcn, _EqualKey, _Alloc > &__hs1, const hash_multiset< ↵
_Val, _HashFcn, _EqualKey, _Alloc > &__hs2)`
- `template<class _Key, class _Tp, class _HF, class _EqKey, class _Alloc >`
`bool operator!= (const hash_multimap< _Key, _Tp, _HF, _EqKey, _Alloc > &__hm1, const hash_multimap< ↵
_Key, _Tp, _HF, _EqKey, _Alloc > &__hm2)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator!= (const Pointer_adapter< _Tp1 > &__lhs, _Tp2 __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator!= (_Tp1 __lhs, const Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator!= (const Pointer_adapter< _Tp1 > &__lhs, const Pointer_adapter< _Tp2 > &__rhs)`

- `template<typename _Tp >`
`bool operator!= (const _Pointer_adapter< _Tp > &__lhs, int __rhs)`
- `template<typename _Tp >`
`bool operator!= (int __lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<typename _Tp >`
`bool operator!= (const _Pointer_adapter< _Tp > &__lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<class _Val, class _Key, class _HF, class _Ex, class _Eq, class _All >`
`bool operator!= (const hashtable< _Val, _Key, _HF, _Ex, _Eq, _All > &__ht1, const hashtable< _Val, _Key, _HF, _Ex, _Eq, _All > &__ht2)`
- `template<typename _Tp, typename _Poolp >`
`bool operator!= (const __mt_alloc< _Tp, _Poolp > &, const __mt_alloc< _Tp, _Poolp > &)`
- `template<class _Tp, class _Alloc >`
`bool operator!= (const slist< _Tp, _Alloc > &_SL1, const slist< _Tp, _Alloc > &_SL2)`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`bool operator!= (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator< _IteratorR, _Container > &__rhs) noexcept`
- `template<typename _Iterator, typename _Container >`
`bool operator!= (const __normal_iterator< _Iterator, _Container > &__lhs, const __normal_iterator< _Iterator, _Container > &__rhs) noexcept`
- `template<typename _Tp, typename _Cond >`
`bool operator!= (const throw_allocator_base< _Tp, _Cond > &, const throw_allocator_base< _Tp, _Cond > &)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator!= (const bitmap_allocator< _Tp1 > &, const bitmap_allocator< _Tp2 > &) throw ()`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator!= (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator!= (const _CharT * __lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator!= (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const _CharT * __rhs)`
- `template<class _CharT, class _Alloc >`
`bool operator!= (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool operator!= (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool operator!= (const rope< _CharT, _Alloc > &__x, const rope< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool operator!= (const _Rope_char_ptr_proxy< _CharT, _Alloc > &__x, const _Rope_char_ptr_proxy< _CharT, _Alloc > &__y)`
- `template<typename _Cond >`
`throw_value_base< _Cond > operator* (const throw_value_base< _Cond > &__a, const throw_value_base< _Cond > &__b)`
- `template<class _CharT, class _Alloc >`
`_Rope_const_iterator< _CharT, _Alloc > operator+ (const _Rope_const_iterator< _CharT, _Alloc > &__x, ptrdiff_t __n)`
- `template<class _CharT, class _Alloc >`
`_Rope_const_iterator< _CharT, _Alloc > operator+ (ptrdiff_t __n, const _Rope_const_iterator< _CharT, _Alloc > &__x)`
- `template<class _CharT, class _Alloc >`
`_Rope_iterator< _CharT, _Alloc > operator+ (const _Rope_iterator< _CharT, _Alloc > &__x, ptrdiff_t __n)`
- `template<class _CharT, class _Alloc >`
`_Rope_iterator< _CharT, _Alloc > operator+ (ptrdiff_t __n, const _Rope_iterator< _CharT, _Alloc > &__x)`

- [illegible]

- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > & operator+= (rope< _CharT, _Alloc > &__left, _CharT __right)`
- `template<class _CharT, class _Alloc >`
`_Rope_const_iterator< _CharT, _Alloc > operator- (const _Rope_const_iterator< _CharT, _Alloc > &__x, ptrdiff_t __n)`
- `template<class _CharT, class _Alloc >`
`ptrdiff_t operator- (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`_Rope_iterator< _CharT, _Alloc > operator- (const _Rope_iterator< _CharT, _Alloc > &__x, ptrdiff_t __n)`
- `template<class _CharT, class _Alloc >`
`ptrdiff_t operator- (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<typename _Cond >`
`throw_value_base< _Cond > operator- (const throw_value_base< _Cond > &__a, const throw_value_base< _Cond > &__b)`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`auto operator- (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator< _IteratorR, _Container > &__rhs) noexcept-> decltype(__lhs.base()-__rhs.base())`
- `template<typename _Iterator, typename _Container >`
`__normal_iterator< _Iterator, _Container >::difference_type operator- (const __normal_iterator< _Iterator, _Container > &__lhs, const __normal_iterator< _Iterator, _Container > &__rhs) noexcept`
- `template<typename _Value, typename _Int, typename _St >`
`bool operator< (const character< _Value, _Int, _St > &lhs, const character< _Value, _Int, _St > &rhs)`
- `template<class _CharT, class _Alloc >`
`bool operator< (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool operator< (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator< (const _Pointer_adapter< _Tp1 > &__lhs, _Tp2 __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator< (_Tp1 __lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator< (const _Pointer_adapter< _Tp1 > &__lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Cond >`
`bool operator< (const throw_value_base< _Cond > &__a, const throw_value_base< _Cond > &__b)`
- `template<class _Tp, class _Alloc >`
`bool operator< (const slist< _Tp, _Alloc > &_SL1, const slist< _Tp, _Alloc > &_SL2)`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`bool operator< (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator< _IteratorR, _Container > &__rhs) noexcept`
- `template<typename _Iterator, typename _Container >`
`bool operator< (const __normal_iterator< _Iterator, _Container > &__lhs, const __normal_iterator< _Iterator, _Container > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator< (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator< (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator< (const _CharT *__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`

- `template<class _CharT, class _Alloc >`
`bool operator< (const rope< _CharT, _Alloc > &__left, const rope< _CharT, _Alloc > &__right)`
- `template<typename _UIntType, size_t __m, size_t __pos1, size_t __sl1, size_t __sl2, size_t __sr1, size_t __sr2, uint32_t __msk1, uint32_t __msk2, uint32_t __msk3, uint32_t __msk4, uint32_t __parity1, uint32_t __parity2, uint32_t __parity3, uint32_t __parity4, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const __gnu_cxx::simd_fast_mersenne_twister_engine< _UIntType, __m, __pos1, __sl1, __sl2, __sr1, __sr2, __msk1, __msk2, __msk3, __msk4, __parity1, __parity2, __parity3, __parity4 > &__x)`
- `std::ostream & operator<< (std::ostream &os, const annotate_base &__b)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const __gnu_cxx::beta_distribution< _RealType > &__x)`
- `template<typename _CharT, typename _Traits, typename _StoreT >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const _Pointer_adapter< _StoreT > &__p)`
- `template<size_t __Dimen, typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const __gnu_cxx::normal_mv_distribution< __Dimen, _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const rice_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const nakagami_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const pareto_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const k_distribution< _RealType > &__x)`
- `template<class _CharT, class _Traits, class _Alloc >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__o, const rope< __gnu_cxx::basic_string< _CharT, _Alloc > &__r)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const arcsine_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const hoyt_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const __gnu_cxx::triangular_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const __gnu_cxx::von_mises_distribution< _RealType > &__x)`
- `template<typename _UIntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const __gnu_cxx::hypergeometric_distribution< _UIntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const logistic_distribution< _RealType > &__x)`
- `template<std::size_t __Dimen, typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const __gnu_cxx::uniform_on_sphere_distribution< __Dimen, _RealType > &__x)`

- `template<class _CharT, class _Traits, class _Alloc >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__o, const`
`rope< _CharT, _Alloc > &__r)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator<= (const _Pointer_adapter< _Tp1 > &__lhs, _Tp2 __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator<= (_Tp1 __lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator<= (const _Pointer_adapter< _Tp1 > &__lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp >`
`bool operator<= (const _Pointer_adapter< _Tp > &__lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<class _Tp, class _Alloc >`
`bool operator<= (const slist< _Tp, _Alloc > &_SL1, const slist< _Tp, _Alloc > &_SL2)`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`bool operator<= (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator< _IteratorR, _Container > &__rhs) noexcept`
- `template<typename _Iterator, typename _Container >`
`bool operator<= (const __normal_iterator< _Iterator, _Container > &__lhs, const __normal_iterator< _Iterator, _Container > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator<= (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator<= (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const _CharT * __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator<= (const _CharT * __lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<class _CharT, class _Alloc >`
`bool operator<= (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool operator<= (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool operator<= (const rope< _CharT, _Alloc > &__x, const rope< _CharT, _Alloc > &__y)`
- `template<typename _Value, typename _Int, typename _St >`
`bool operator== (const character< _Value, _Int, _St > &lhs, const character< _Value, _Int, _St > &rhs)`
- `template<typename _Tp >`
`bool operator== (const new_allocator< _Tp > &, const new_allocator< _Tp > &)`
- `template<typename _Tp >`
`bool operator== (const malloc_allocator< _Tp > &, const malloc_allocator< _Tp > &)`
- `template<typename _Tp, typename _Array >`
`bool operator== (const array_allocator< _Tp, _Array > &, const array_allocator< _Tp, _Array > &)`
- `template<typename _Tp >`
`bool operator== (const __pool_alloc< _Tp > &, const __pool_alloc< _Tp > &)`
- `template<class _Val, class _Key, class _HF, class _Ex, class _Eq, class _All >`
`bool operator== (const hashtable< _Val, _Key, _HF, _Ex, _Eq, _All > &__ht1, const hashtable< _Val, _Key, _HF, _Ex, _Eq, _All > &__ht2)`
- `template<class _Value, class _HashFcn, class _EqualKey, class _Alloc >`
`bool operator== (const hash_set< _Value, _HashFcn, _EqualKey, _Alloc > &__hs1, const hash_set< _Value, _HashFcn, _EqualKey, _Alloc > &__hs2)`
- `template<class _Key, class _Tp, class _HashFn, class _EqKey, class _Alloc >`
`bool operator== (const hash_map< _Key, _Tp, _HashFn, _EqKey, _Alloc > &__hm1, const hash_map< _Key, _Tp, _HashFn, _EqKey, _Alloc > &__hm2)`

- `template<typename _UIntType, size_t __m, size_t __pos1, size_t __sl1, size_t __sl2, size_t __sr1, size_t __sr2, uint32_t __msk1, uint32_t __msk2, uint32_t __msk3, uint32_t __msk4, uint32_t __parity1, uint32_t __parity2, uint32_t __parity3, uint32_t __parity4>`
`bool operator== (const __gnu_cxx::simd_fast_mersenne_twister_engine< _UIntType, __m, __pos1, __sl1, __sl2, __sr1, __sr2, __msk1, __msk2, __msk3, __msk4, __parity1, __parity2, __parity3, __parity4 > &__lhs, const __gnu_cxx::simd_fast_mersenne_twister_engine< _UIntType, __m, __pos1, __sl1, __sl2, __sr1, __sr2, __msk1, __msk2, __msk3, __msk4, __parity1, __parity2, __parity3, __parity4 > &__rhs)`
- `template<class _CharT, class _Alloc >`
`bool operator== (const _Rope_char_ptr_proxy< _CharT, _Alloc > &__x, const _Rope_char_ptr_proxy< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool operator== (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool operator== (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<class _Val, class _HashFcn, class _EqualKey, class _Alloc >`
`bool operator== (const hash_multiset< _Val, _HashFcn, _EqualKey, _Alloc > &__hs1, const hash_multiset< _Val, _HashFcn, _EqualKey, _Alloc > &__hs2)`
- `template<class _Key, class _Tp, class _HF, class _EqKey, class _Alloc >`
`bool operator== (const hash_multimap< _Key, _Tp, _HF, _EqKey, _Alloc > &__hm1, const hash_multimap< _Key, _Tp, _HF, _EqKey, _Alloc > &__hm2)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator== (const _Pointer_adapter< _Tp1 > &__lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator== (const _Pointer_adapter< _Tp1 > &__lhs, _Tp2 __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator== (_Tp1 __lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp >`
`bool operator== (const _Pointer_adapter< _Tp > &__lhs, int __rhs)`
- `template<typename _Tp >`
`bool operator== (int __lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<typename _Tp >`
`bool operator== (const _Pointer_adapter< _Tp > &__lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<size_t _Dimen, typename _RealType >`
`bool operator== (const __gnu_cxx::normal_mv_distribution< _Dimen, _RealType > &__d1, const __gnu_cxx::normal_mv_distribution< _Dimen, _RealType > &__d2)`
- `template<typename _Cond >`
`bool operator== (const throw_value_base< _Cond > &__a, const throw_value_base< _Cond > &__b)`
- `template<typename _Tp, typename _Poolp >`
`bool operator== (const __mt_alloc< _Tp, _Poolp > &, const __mt_alloc< _Tp, _Poolp > &)`
- `template<class _Tp, class _Alloc >`
`bool operator== (const slist< _Tp, _Alloc > &__SL1, const slist< _Tp, _Alloc > &__SL2)`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`bool operator== (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator< _IteratorR, _Container > &__rhs) noexcept`
- `template<typename _Iterator, typename _Container >`
`bool operator== (const __normal_iterator< _Iterator, _Container > &__lhs, const __normal_iterator< _Iterator, _Container > &__rhs) noexcept`
- `template<typename _Tp, typename _Cond >`
`bool operator== (const throw_allocator_base< _Tp, _Cond > &, const throw_allocator_base< _Tp, _Cond > &)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator== (const bitmap_allocator< _Tp1 > &, const bitmap_allocator< _Tp2 > &) throw ()`

- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator== (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const __versa_string< _CharT,`
`_Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, template< typename, typename, typename > class _Base>`
`__enable_if< std::is_char< _CharT >::value, bool >::type operator== (const __versa_string< _CharT,`
`std::char_traits< _CharT >, std::allocator< _CharT >, _Base > &__lhs, const __versa_string< _CharT, std::`
`char_traits< _CharT >, std::allocator< _CharT >, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator== (const _CharT *__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator== (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const _CharT *__rhs)`
- `template<class _CharT, class _Alloc >`
`bool operator== (const rope< _CharT, _Alloc > &__left, const rope< _CharT, _Alloc > &__right)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator> (const _Pointer_adapter< _Tp1 > &__lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator> (_Tp1 __lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator> (const _Pointer_adapter< _Tp1 > &__lhs, _Tp2 __rhs)`
- `template<typename _Tp >`
`bool operator> (const _Pointer_adapter< _Tp > &__lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<class _Tp, class _Alloc >`
`bool operator> (const slist< _Tp, _Alloc > &__SL1, const slist< _Tp, _Alloc > &__SL2)`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`bool operator> (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator< _`
`IteratorR, _Container > &__rhs) noexcept`
- `template<typename _Iterator, typename _Container >`
`bool operator> (const __normal_iterator< _Iterator, _Container > &__lhs, const __normal_iterator< _Iterator,`
`_Container > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator> (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const __versa_string< _CharT,`
`_Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator> (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator> (const _CharT *__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<class _CharT, class _Alloc >`
`bool operator> (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT,`
`_Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool operator> (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool operator> (const rope< _CharT, _Alloc > &__x, const rope< _CharT, _Alloc > &__y)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator>= (const _Pointer_adapter< _Tp1 > &__lhs, _Tp2 __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator>= (_Tp1 __lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator>= (const _Pointer_adapter< _Tp1 > &__lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp >`
`bool operator>= (const _Pointer_adapter< _Tp > &__lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<class _Tp, class _Alloc >`
`bool operator>= (const slist< _Tp, _Alloc > &__SL1, const slist< _Tp, _Alloc > &__SL2)`

- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`bool operator>= (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator< _IteratorR, _Container > &__rhs) noexcept`
- `template<typename _Iterator, typename _Container >`
`bool operator>= (const __normal_iterator< _Iterator, _Container > &__lhs, const __normal_iterator< _Iterator, _Container > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator>= (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator>= (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const _CharT * __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool operator>= (const _CharT * __lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<class _CharT, class _Alloc >`
`bool operator>= (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool operator>= (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool operator>= (const rope< _CharT, _Alloc > &__x, const rope< _CharT, _Alloc > &__y)`
- `template<typename _UIntType, size_t __m, size_t __pos1, size_t __sl1, size_t __sl2, size_t __sr1, size_t __sr2, uint32_t __msk1, uint32_t __msk2, uint32_t __msk3, uint32_t __msk4, uint32_t __parity1, uint32_t __parity2, uint32_t __parity3, uint32_t __parity4, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, __gnu_cxx::simd_fast_mersenne_twister_engine< _UIntType, __m, __pos1, __sl1, __sl2, __sr1, __sr2, __msk1, __msk2, __msk3, __msk4, __parity1, __parity2, __parity3, __parity4 > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, __gnu_cxx::beta_distribution< _RealType > & __x)`
- `template<size_t __Dimen, typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, __gnu_cxx::normal_mv_distribution< __Dimen, _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, rice_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, nakagami_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, pareto_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, k_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, arcsine_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, hoyt_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, __gnu_cxx::triangular_distribution< _RealType > & __x)`

- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, __gnu_cxx::von_mises_distribution< _RealType > & __x)`
- `template<typename _UIntType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, __gnu_cxx::hypergeometric_distribution< _UIntType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, logistic_distribution< _RealType > & __x)`
- `template<std::size_t _Dimen, typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, __gnu_cxx::uniform_on_sphere_distribution< _Dimen, _RealType > & __x)`
- `template<typename _Tp, typename _Integer, typename _MonoidOperation >`
`_Tp power (_Tp __x, _Integer __n, _MonoidOperation __monoid_op)`
- `template<typename _Tp, typename _Integer >`
`_Tp power (_Tp __x, _Integer __n)`
- `template<typename _InputIterator, typename _RandomAccessIterator >`
`_RandomAccessIterator random_sample (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out_first, _RandomAccessIterator __out_last)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _RandomNumberGenerator >`
`_RandomAccessIterator random_sample (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out_first, _RandomAccessIterator __out_last, _RandomNumberGenerator & __rand)`
- `template<typename _ForwardIterator, typename _OutputIterator, typename _Distance >`
`_OutputIterator random_sample_n (_ForwardIterator __first, _ForwardIterator __last, _OutputIterator __out, const _Distance __n)`
- `template<typename _ForwardIterator, typename _OutputIterator, typename _Distance, typename _RandomNumberGenerator >`
`_OutputIterator random_sample_n (_ForwardIterator __first, _ForwardIterator __last, _OutputIterator __out, const _Distance __n, _RandomNumberGenerator & __rand)`
- `void rotate (_Rope_iterator< char, __STL_DEFAULT_ALLOCATOR(char)> __first, _Rope_iterator< char, __STL_DEFAULT_ALLOCATOR(char)> __middle, _Rope_iterator< char, __STL_DEFAULT_ALLOCATOR(char)> __last)`
- `template<typename _Tp >`
`void swap (_ExtPtr_allocator< _Tp > & __larg, _ExtPtr_allocator< _Tp > & __rarg)`
- `template<class _Val, class _HashFcn, class _EqualKey, class _Alloc >`
`void swap (hash_set< _Val, _HashFcn, _EqualKey, _Alloc > & __hs1, hash_set< _Val, _HashFcn, _EqualKey, _Alloc > & __hs2)`
- `template<class _Key, class _Tp, class _HashFcn, class _EqKey, class _Alloc >`
`void swap (hash_map< _Key, _Tp, _HashFcn, _EqKey, _Alloc > & __hm1, hash_map< _Key, _Tp, _HashFcn, _EqKey, _Alloc > & __hm2)`
- `template<class _Val, class _HashFcn, class _EqualKey, class _Alloc >`
`void swap (hash_multiset< _Val, _HashFcn, _EqualKey, _Alloc > & __hs1, hash_multiset< _Val, _HashFcn, _EqualKey, _Alloc > & __hs2)`
- `template<class _Key, class _Tp, class _HashFcn, class _EqKey, class _Alloc >`
`void swap (hash_multimap< _Key, _Tp, _HashFcn, _EqKey, _Alloc > & __hm1, hash_multimap< _Key, _Tp, _HashFcn, _EqKey, _Alloc > & __hm2)`
- `template<typename _Cond >`
`void swap (throw_value_base< _Cond > & __a, throw_value_base< _Cond > & __b)`
- `template<class _Val, class _Key, class _HF, class _Extract, class _EqKey, class _All >`
`void swap (hashtable< _Val, _Key, _HF, _Extract, _EqKey, _All > & __ht1, hashtable< _Val, _Key, _HF, _Extract, _EqKey, _All > & __ht2)`
- `template<class _Tp, class _Alloc >`
`void swap (slist< _Tp, _Alloc > & __x, slist< _Tp, _Alloc > & __y)`

- `template<class _CharT, class __Alloc >`
`void swap (_Rope_char_ref_proxy< _CharT, __Alloc > __a, _Rope_char_ref_proxy< _CharT, __Alloc > __b)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`void swap (__versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<class _CharT, class __Alloc >`
`void swap (rope< _CharT, _Alloc > &__x, rope< _CharT, _Alloc > &__y)`
- `_Atomic_word int throw ()`
- `template<typename _InputIter, typename _Size, typename _ForwardIter >`
`pair< _InputIter, _ForwardIter > uninitialized_copy_n (InputIter __first, _Size __count, _ForwardIter __result)`

Variables

- `static const _Lock_policy __default_lock_policy`
- `static _Atomic_word int __val`
- `template<class _CharT, class __Alloc >`
`rope< _CharT, _Alloc > identity_element (_Rope_Concat_fn< _CharT, _Alloc >)`

4.1.1 Detailed Description

GNU extensions for public use.

4.1.2 Function Documentation

4.1.2.1 `template<typename _ToType, typename _FromType > _ToType __gnu_cxx::__static_pointer_cast (const _FromType & __arg) [inline]`

Casting operations for cases where `_FromType` is not a standard pointer. `_ToType` can be a standard or non-standard pointer. Given that `_FromType` is not a pointer, it must have a `get()` method that returns the standard pointer equivalent of the address it points to, and must have an `element_type` typedef which names the type it points to.

Definition at line 68 of file `cast.h`.

4.1.2.2 `template<typename _ToType, typename _FromType > _ToType __gnu_cxx::__static_pointer_cast (_FromType * __arg) [inline]`

Casting operations for cases where `_FromType` is a standard pointer. `_ToType` can be a standard or non-standard pointer.

Definition at line 96 of file `cast.h`.

4.1.2.3 `size_t __gnu_cxx::Bit_scan_forward (size_t __num) [inline]`

Generic Version of the `bsf` instruction.

Definition at line 513 of file `bitmap_allocator.h`.

Referenced by `__gnu_cxx::bitmap_allocator< _Tp >::M_allocate_single_object()`.

4.1.2.4 `template<typename _Tpa, typename _Tpc, typename _Tp> __gnu_cxx::__promote_3<_Tpa, _Tpc, _Tp>::__type
__gnu_cxx::conf_hyperg(_Tpa __a, _Tpc __c, _Tp __x) [inline]`

Return the confluent hypergeometric function ${}_1F_1(a; c; x)$ of real numeratorial parameter a , denominatorial parameter c , and argument x .

The confluent hypergeometric function is defined by

$${}_1F_1(a; c; x) = \sum_{n=0}^{\infty} \frac{(a)_n x^n}{(c)_n n!}$$

where the Pochhammer symbol is $(x)_k = (x)(x+1)\dots(x+k-1)$, $(x)_0 = 1$

Parameters

<code>__a</code>	The numeratorial parameter
<code>__c</code>	The denominatorial parameter
<code>__x</code>	The argument

Definition at line 1249 of file `specfun.h`.

4.1.2.5 `float __gnu_cxx::conf_hypergf(float __a, float __c, float __x) [inline]`

Return the confluent hypergeometric function ${}_1F_1(a; c; x)$ of `float` numeratorial parameter a , denominatorial parameter c , and argument x .

See also

`conf_hyperg` for details.

Definition at line 1217 of file `specfun.h`.

4.1.2.6 `long double __gnu_cxx::conf_hypergl(long double __a, long double __c, long double __x) [inline]`

Return the confluent hypergeometric function ${}_1F_1(a; c; x)$ of `long double` numeratorial parameter a , denominatorial parameter c , and argument x .

See also

`conf_hyperg` for details.

Definition at line 1228 of file `specfun.h`.

4.1.2.7 `template<typename _Tpa, typename _Tpb, typename _Tpc, typename _Tp> __gnu_cxx::__promote_4<_Tpa, _Tpb, _Tpc, _Tp>::__type __gnu_cxx::hyperg(_Tpa __a, _Tpb __b, _Tpc __c, _Tp __x) [inline]`

Return the hypergeometric function ${}_2F_1(a, b; c; x)$ of real numeratorial parameters a and b , denominatorial parameter c , and argument x .

The hypergeometric function is defined by

$${}_2F_1(a; c; x) = \sum_{n=0}^{\infty} \frac{(a)_n (b)_n x^n}{(c)_n n!}$$

where the Pochhammer symbol is $(x)_k = (x)(x+1)\dots(x+k-1)$, $(x)_0 = 1$

Parameters

$_a$	The first numeratorial parameter
$_b$	The second numeratorial parameter
$_c$	The denominatorial parameter
$_x$	The argument

Definition at line 1298 of file specfun.h.

4.1.2.8 `float __gnu_cxx::hypergf (float $_a$, float $_b$, float $_c$, float $_x$)` `[inline]`

Return the hypergeometric function ${}_2F_1(a, b; c; x)$ of @ float numeratorial parameters a and b, denominatorial parameter c, and argument x.

See also

hyperg for details.

Definition at line 1265 of file specfun.h.

4.1.2.9 `long double __gnu_cxx::hypergl (long double $_a$, long double $_b$, long double $_c$, long double $_x$)` `[inline]`

Return the hypergeometric function ${}_2F_1(a, b; c; x)$ of long double numeratorial parameters a and b, denominatorial parameter c, and argument x.

See also

hyperg for details.

Definition at line 1276 of file specfun.h.

4.1.2.10 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> bool __gnu_cxx::operator!=(const __versa_string< _CharT, _Traits, _Alloc, _Base > & $_lhs$, const __versa_string< _CharT, _Traits, _Alloc, _Base > & $_rhs$)` `[inline]`

Test difference of two strings.

Parameters

$_lhs$	First string.
$_rhs$	Second string.

Returns

True if `__lhs.compare(__rhs) != 0`. False otherwise.

Definition at line 2388 of file `vstring.h`.

```
4.1.2.11 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> bool __gnu_cxx::operator!=( const _CharT * __lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &
__rhs ) [inline]
```

Test difference of C string and string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__rhs.compare(__lhs) != 0`. False otherwise.

Definition at line 2401 of file `vstring.h`.

```
4.1.2.12 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> bool __gnu_cxx::operator!=( const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs, const _CharT *
__rhs ) [inline]
```

Test difference of string and C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs.compare(__rhs) != 0`. False otherwise.

Definition at line 2414 of file `vstring.h`.

```
4.1.2.13 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> __versa_string< _CharT, _Traits, _Alloc, _Base > __gnu_cxx::operator+( const __versa_string< _CharT,
_Traits, _Alloc, _Base > & __lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs )
```

Concatenate two strings.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Last string.

Returns

New string with value of `__lhs` followed by `__rhs`.

Definition at line 181 of file `vstring.tcc`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::append()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::reserve()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

4.1.2.14 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string<_CharT, _Traits, _Alloc, _Base> __gnu_cxx::operator+ (const _CharT * __lhs, const __versa_string<_CharT, _Traits, _Alloc, _Base> & __rhs)`

Concatenate C string and string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Last string.

Returns

New string with value of `__lhs` followed by `__rhs`.

Definition at line 194 of file `vstring.tcc`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

4.1.2.15 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string<_CharT, _Traits, _Alloc, _Base> __gnu_cxx::operator+ (_CharT __lhs, const __versa_string<_CharT, _Traits, _Alloc, _Base> & __rhs)`

Concatenate character and string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Last string.

Returns

New string with `__lhs` followed by `__rhs`.

Definition at line 211 of file `vstring.tcc`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::append()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::push_back()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::reserve()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

4.1.2.16 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string<_CharT, _Traits, _Alloc, _Base> __gnu_cxx::operator+ (const __versa_string<_CharT, _Traits, _Alloc, _Base> & __lhs, const _CharT * __rhs)`

Concatenate string and C string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Last string.

Returns

New string with `__lhs` followed by `__rhs`.

Definition at line 224 of file `vstring.tcc`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

4.1.2.17 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string<_CharT, _Traits, _Alloc, _Base> __gnu_cxx::operator+ (const __versa_string<_CharT, _Traits, _Alloc, _Base> & __lhs, _CharT __rhs)`

Concatenate string and character.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Last string.

Returns

New string with `__lhs` followed by `__rhs`.

Definition at line 241 of file `vstring.tcc`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::append()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::copy()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::push_back()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::reserve()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

4.1.2.18 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> bool __gnu_cxx::operator< (const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs) [inline]`

Test if string precedes string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs` precedes `__rhs`. False otherwise.

Definition at line 2428 of file `vstring.h`.

References `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::compare()`.

4.1.2.19 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> bool __gnu_cxx::operator< (const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs, const _CharT * __rhs) [inline]`

Test if string precedes C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs` precedes `__rhs`. False otherwise.

Definition at line 2441 of file `vstring.h`.

4.1.2.20 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> bool __gnu_cxx::operator< (const _CharT * __lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs) [inline]`

Test if C string precedes string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__lhs` precedes `__rhs`. False otherwise.

Definition at line 2454 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare()`.

4.1.2.21 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> bool __gnu_cxx::operator<= (const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs) [inline]`

Test if string doesn't follow string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs` doesn't follow `__rhs`. False otherwise.

Definition at line 2508 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare()`.

4.1.2.22 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> bool __gnu_cxx::operator<= (const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs, const _CharT * __rhs) [inline]`

Test if string doesn't follow C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs` doesn't follow `__rhs`. False otherwise.

Definition at line 2521 of file `vstring.h`.

4.1.2.23 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> bool __gnu_cxx::operator<= (const _CharT * __lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs) [inline]`

Test if C string doesn't follow string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__lhs` doesn't follow `__rhs`. False otherwise.

Definition at line 2534 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::compare()`.

4.1.2.24 `template<typename _Tp > bool __gnu_cxx::operator==(const _Pointer_adapter<_Tp > & __lhs, const _Pointer_adapter<_Tp > & __rhs) [inline]`

Comparison operators for `_Pointer_adapter` defer to the base class' comparison operators, when possible.

Definition at line 529 of file `pointer.h`.

References `std::addressof()`.

4.1.2.25 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> bool __gnu_cxx::operator==(const __versa_string<_CharT, _Traits, _Alloc, _Base > & __lhs, const __versa_string<_CharT, _Traits, _Alloc, _Base > & __rhs) [inline]`

Test equivalence of two strings.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs.compare(__rhs) == 0`. False otherwise.

Definition at line 2337 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::compare()`.

4.1.2.26 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> bool __gnu_cxx::operator==(const _CharT * __lhs, const __versa_string<_CharT, _Traits, _Alloc, _Base > & __rhs) [inline]`

Test equivalence of C string and string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__rhs.compare(__lhs) == 0`. False otherwise.

Definition at line 2361 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare()`.

4.1.2.27 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> bool __gnu_cxx::operator==(const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs, const _CharT * __rhs) [inline]`

Test equivalence of string and C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs.compare(__rhs) == 0`. False otherwise.

Definition at line 2374 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare()`.

4.1.2.28 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> bool __gnu_cxx::operator> (const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > & __rhs) [inline]`

Test if string follows string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs` follows `__rhs`. False otherwise.

Definition at line 2468 of file vstring.h.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::compare()`.

4.1.2.29 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> bool __gnu_cxx::operator> (const __versa_string<_CharT, _Traits, _Alloc, _Base > & __lhs, const _CharT * __rhs) [inline]`

Test if string follows C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs` follows `__rhs`. False otherwise.

Definition at line 2481 of file vstring.h.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::compare()`.

4.1.2.30 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> bool __gnu_cxx::operator> (const _CharT * __lhs, const __versa_string<_CharT, _Traits, _Alloc, _Base > & __rhs) [inline]`

Test if C string follows string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__lhs` follows `__rhs`. False otherwise.

Definition at line 2494 of file vstring.h.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::compare()`.

4.1.2.31 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> bool __gnu_cxx::operator>= (const __versa_string<_CharT, _Traits, _Alloc, _Base > & __lhs, const __versa_string<_CharT, _Traits, _Alloc, _Base > & __rhs) [inline]`

Test if string doesn't precede string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs` doesn't precede `__rhs`. False otherwise.

Definition at line 2548 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::compare()`.

```
4.1.2.32 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> bool __gnu_cxx::operator>= ( const __versa_string< _CharT, _Traits, _Alloc, _Base > & __lhs, const _CharT
* __rhs ) [inline]
```

Test if string doesn't precede C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs` doesn't precede `__rhs`. False otherwise.

Definition at line 2561 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::compare()`.

```
4.1.2.33 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> bool __gnu_cxx::operator>= ( const _CharT * __lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base >
& __rhs ) [inline]
```

Test if C string doesn't precede string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__lhs` doesn't precede `__rhs`. False otherwise.

Definition at line 2574 of file vstring.h.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::compare()`.

4.1.2.34 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> void __gnu_cxx::swap (__versa_string<_CharT, _Traits, _Alloc, _Base > &__lhs, __versa_string<_CharT, _Traits, _Alloc, _Base > &__rhs) [inline]`

Swap contents of two strings.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Exchanges the contents of `__lhs` and `__rhs` in constant time.

Definition at line 2588 of file vstring.h.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::swap()`.

4.2 `__gnu_cxx::__detail` Namespace Reference

Classes

- class [__mini_vector](#)
- class [_Bitmap_counter](#)
- class [_Ffit_finder](#)

Enumerations

- enum { `_S_max_ropes_depth` }
- enum { `bits_per_byte`, `bits_per_block` }
- enum `_Tag` { `_S_leaf`, `_S_concat`, `_S_substringfn`, `_S_function` }

Functions

- void [__bit_allocate](#) (size_t *__pmap, size_t __pos) throw ()
- void [__bit_free](#) (size_t *__pmap, size_t __pos) throw ()
- template<typename _ForwardIterator, typename _Tp, typename _Compare >
_ForwardIterator **lower_bound** (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)
- template<typename _AddrPair >
size_t [__num_bitmaps](#) (_AddrPair __ap)
- template<typename _AddrPair >
size_t [__num_blocks](#) (_AddrPair __ap)

4.2.1 Detailed Description

Implementation details not part of the namespace `__gnu_cxx` interface.

4.2.2 Function Documentation

4.2.2.1 `void __gnu_cxx::__detail::__bit_allocate (size_t * __pmap, size_t __pos) throw` `[inline]`

Mark a memory address as allocated by re-setting the corresponding bit in the bit-map.

Definition at line 488 of file `bitmap_allocator.h`.

Referenced by `__gnu_cxx::bitmap_allocator<_Tp>::_M_allocate_single_object()`.

4.2.2.2 `void __gnu_cxx::__detail::__bit_free (size_t * __pmap, size_t __pos) throw` `[inline]`

Mark a memory address as free by setting the corresponding bit in the bit-map.

Definition at line 499 of file `bitmap_allocator.h`.

Referenced by `__gnu_cxx::bitmap_allocator<_Tp>::_M_deallocate_single_object()`.

4.2.2.3 `template<typename _AddrPair> size_t __gnu_cxx::__detail::__num_bitmaps (_AddrPair __ap)` `[inline]`

The number of Bit-maps pointed to by the address pair passed to the function.

Definition at line 276 of file `bitmap_allocator.h`.

References `__num_blocks()`.

Referenced by `__gnu_cxx::bitmap_allocator<_Tp>::_M_allocate_single_object()`, and `__gnu_cxx::bitmap_allocator<_Tp>::_M_deallocate_single_object()`.

4.2.2.4 `template<typename _AddrPair> size_t __gnu_cxx::__detail::__num_blocks (_AddrPair __ap)` `[inline]`

The number of Blocks pointed to by the address pair passed to the function.

Definition at line 268 of file `bitmap_allocator.h`.

Referenced by `__num_bitmaps()`.

4.3 `__gnu_cxx::typelist` Namespace Reference

Functions

- `template<typename Fn , typename Typelist >`
`void apply (Fn &, Typelist)`
- `template<typename Gn , typename Typelist >`
`void apply_generator (Gn &, Typelist)`
- `template<typename Gn , typename TypelistT , typename TypelistV >`
`void apply_generator (Gn &, TypelistT, TypelistV)`
- `template<typename Fn , typename Typelist >`
`void apply_generator (Fn &fn, Typelist)`
- `template<typename Fn , typename TypelistT , typename TypelistV >`
`void apply_generator (Fn &fn, TypelistT, TypelistV)`

4.3.1 Detailed Description

GNU typelist extensions for public compile-time use.

4.3.2 Function Documentation

4.3.2.1 `template<typename Gn , typename Typelist > void __gnu_cxx::typelist::apply_generator (Gn &, Typelist)`

Apply all typelist types to generator functor.

4.4 `__gnu_debug` Namespace Reference

Classes

- class `_After_nth_from`
- struct `_BeforeBeginHelper`
- class `_Equal_to`
- class `_Not_equal_to`
- class `_Safe_container`
- class `_Safe_forward_list`
- class `_Safe_iterator`
- class `_Safe_iterator_base`
- class `_Safe_local_iterator`
- class `_Safe_local_iterator_base`
- class `_Safe_node_sequence`
- class `_Safe_sequence`
- class `_Safe_sequence_base`
- class `_Safe_unordered_container`
- class `_Safe_unordered_container_base`
- class `_Safe_vector`
- struct `_Sequence_traits`
- class `basic_string`

Typedefs

- typedef `basic_string`< char > `string`
- typedef `basic_string`< wchar_t > `wstring`

Enumerations

- enum `_Debug_msg_id` {
`__msg_valid_range`, `__msg_insert_singular`, `__msg_insert_different`, `__msg_erase_bad`,
`__msg_erase_different`, `__msg_subscript_oob`, `__msg_empty`, `__msg_unpartitioned`,
`__msg_unpartitioned_pred`, `__msg_unsorted`, `__msg_unsorted_pred`, `__msg_not_heap`,
`__msg_not_heap_pred`, `__msg_bad_bitset_write`, `__msg_bad_bitset_read`, `__msg_bad_bitset_flip`,
`__msg_self_splice`, `__msg_splice_alloc`, `__msg_splice_bad`, `__msg_splice_other`,
`__msg_splice_overlap`, `__msg_init_singular`, `__msg_init_copy_singular`, `__msg_init_const_singular`,
`__msg_copy_singular`, `__msg_bad_deref`, `__msg_bad_inc`, `__msg_bad_dec`,
`__msg_iter_subscript_oob`, `__msg_advance_oob`, `__msg_retreat_oob`, `__msg_iter_compare_bad`,
`__msg_compare_different`, `__msg_iter_order_bad`, `__msg_order_different`, `__msg_distance_bad`,
`__msg_distance_different`, `__msg_deref_istream`, `__msg_inc_istream`, `__msg_output_ostream`,
`__msg_deref_istreambuf`, `__msg_inc_istreambuf`, `__msg_insert_after_end`, `__msg_erase_after_bad`,
`__msg_valid_range2`, `__msg_local_iter_compare_bad`, `__msg_non_empty_range`, `__msg_self_move_`↵
`assign`,
`__msg_bucket_index_oob`, `__msg_valid_load_factor`, `__msg_equal_allocs`, `__msg_insert_range_from_`↵
`__self`,
`__msg_irreflexive_ordering` }
- enum `_Distance_precision` { `__dp_none`, `__dp_equality`, `__dp_sign`, `__dp_exact` }

Functions

- template<typename `_Iterator` >
`auto __base` (const `std::reverse_iterator`< `_Iterator` > &`__it`) -> `decltype`(`std::__make_reverse_iterator`(↵
`base`(`__it`.`base`())))
- template<typename `_Iterator` >
`auto __base` (const `std::move_iterator`< `_Iterator` > &`__it`) -> `decltype`(`std::make_move_iterator`(↵
`base`(`__it`.↵
`base`())))
- template<typename `_Iterator` >
`_Iterator __base` (`_Iterator` `__it`)
- template<typename `_Iterator`, typename `_Sequence` >
`_Iterator __base` (const `_Safe_iterator`< `_Iterator`, `_Sequence` > &`__it`, `std::random_access_iterator_tag`)
- template<typename `_Iterator`, typename `_Sequence` >
`const _Safe_iterator`< `_Iterator`, `_Sequence` > & `__base` (const `_Safe_iterator`< `_Iterator`, `_Sequence` > &`__it`,
`std::input_iterator_tag`)
- template<typename `_Iterator`, typename `_Sequence` >
`auto __base` (const `_Safe_iterator`< `_Iterator`, `_Sequence` > &`__it`) -> `decltype`(↵
`base`(`__it`, `std::__iterator_`↵
`category`(`__it`)))
- template<typename `_Iterator` >
`bool __check_dereferenceable` (const `_Iterator` &)
- template<typename `_Tp` >
`bool __check_dereferenceable` (const `_Tp` *`__ptr`)
- template<typename `_Iterator`, typename `_Sequence` >
`bool __check_dereferenceable` (const `_Safe_local_iterator`< `_Iterator`, `_Sequence` > &`__x`)

- `template<typename _Iterator, typename _Sequence >`
`bool __check_dereferenceable (const __Safe_iterator< _Iterator, _Sequence > &__x)`
- `template<typename _ForwardIterator, typename _Tp >`
`bool __check_partitioned_lower (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value)`
- `template<typename _ForwardIterator, typename _Tp, typename _Pred >`
`bool __check_partitioned_lower (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value, _Pred __pred)`
- `template<typename _ForwardIterator, typename _Tp >`
`bool __check_partitioned_upper (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value)`
- `template<typename _ForwardIterator, typename _Tp, typename _Pred >`
`bool __check_partitioned_upper (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value, _Pred __pred)`
- `template<typename _Iterator >`
`bool __check_singular (const _Iterator &)`
- `template<typename _Tp >`
`bool __check_singular (const _Tp * __ptr)`
- `bool __check_singular_aux (const void *)`
- `bool __check_singular_aux (const __Safe_iterator_base * __x)`
- `template<typename _InputIterator >`
`bool __check_sorted (const _InputIterator &__first, const _InputIterator &__last)`
- `template<typename _InputIterator, typename _Predicate >`
`bool __check_sorted (const _InputIterator &__first, const _InputIterator &__last, _Predicate __pred)`
- `template<typename _InputIterator >`
`bool __check_sorted_aux (const _InputIterator &, const _InputIterator &, std::input_iterator_tag)`
- `template<typename _ForwardIterator >`
`bool __check_sorted_aux (_ForwardIterator __first, _ForwardIterator __last, std::forward_iterator_tag)`
- `template<typename _InputIterator, typename _Predicate >`
`bool __check_sorted_aux (const _InputIterator &, const _InputIterator &, _Predicate, std::input_iterator_tag)`
- `template<typename _ForwardIterator, typename _Predicate >`
`bool __check_sorted_aux (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred, std::forward_iterator_tag)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`bool __check_sorted_set (const _InputIterator1 &__first, const _InputIterator1 &__last, const _InputIterator2 &)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Predicate >`
`bool __check_sorted_set (const _InputIterator1 &__first, const _InputIterator1 &__last, const _InputIterator2 &, _Predicate __pred)`
- `template<typename _InputIterator >`
`bool __check_sorted_set_aux (const _InputIterator &__first, const _InputIterator &__last, std::__true_type)`
- `template<typename _InputIterator >`
`bool __check_sorted_set_aux (const _InputIterator &, const _InputIterator &, std::__false_type)`
- `template<typename _InputIterator, typename _Predicate >`
`bool __check_sorted_set_aux (const _InputIterator &__first, const _InputIterator &__last, _Predicate __pred, std::__true_type)`
- `template<typename _InputIterator, typename _Predicate >`
`bool __check_sorted_set_aux (const _InputIterator &, const _InputIterator &, _Predicate, std::__false_type)`
- `template<typename _CharT, typename _Integer >`
`const _CharT * __check_string (const _CharT * __s, const _Integer & __n __attribute__((__unused__)))`
- `template<typename _CharT >`
`const _CharT * __check_string (const _CharT * __s)`
- `template<typename _InputIterator >`
`_InputIterator __check_valid_range (const _InputIterator &__first, const _InputIterator &__last __attribute__((__unused__)))`

- `template<typename _Iterator, typename _Sequence, typename _InputIterator >`
`bool __foreign_iterator (const __Safe_iterator< _Iterator, _Sequence > &__it, _InputIterator __other, _InputIterator __other_end)`
- `template<typename _Iterator, typename _Sequence, typename _Integral >`
`bool __foreign_iterator_aux (const __Safe_iterator< _Iterator, _Sequence > &, _Integral, _Integral, std::__true_type)`
- `template<typename _Iterator, typename _Sequence, typename _InputIterator >`
`bool __foreign_iterator_aux (const __Safe_iterator< _Iterator, _Sequence > &__it, _InputIterator __other, _InputIterator __other_end, std::__false_type)`
- `template<typename _Iterator, typename _Sequence, typename _OtherIterator >`
`bool __foreign_iterator_aux2 (const __Safe_iterator< _Iterator, _Sequence > &__it, const __Safe_iterator< _OtherIterator, _Sequence > &__other, const __Safe_iterator< _OtherIterator, _Sequence > &)`
- `template<typename _Iterator, typename _Sequence, typename _OtherIterator, typename _OtherSequence >`
`bool __foreign_iterator_aux2 (const __Safe_iterator< _Iterator, _Sequence > &__it, const __Safe_iterator< _OtherIterator, _OtherSequence > &, const __Safe_iterator< _OtherIterator, _OtherSequence > &)`
- `template<typename _Iterator, typename _Sequence, typename _InputIterator >`
`bool __foreign_iterator_aux2 (const __Safe_iterator< _Iterator, _Sequence > &__it, const _InputIterator &__other, const _InputIterator &__other_end)`
- `template<typename _Iterator, typename _Sequence, typename _InputIterator >`
`bool __foreign_iterator_aux3 (const __Safe_iterator< _Iterator, _Sequence > &__it, const _InputIterator &__other, const _InputIterator &__other_end, std::__true_type)`
- `template<typename _Iterator, typename _Sequence, typename _InputIterator >`
`bool __foreign_iterator_aux3 (const __Safe_iterator< _Iterator, _Sequence > &, const _InputIterator &, const _InputIterator &, std::__false_type)`
- `template<typename _Iterator, typename _Sequence >`
`bool __foreign_iterator_aux4 (const __Safe_iterator< _Iterator, _Sequence > &__it, const typename _Sequence::value_type *__other)`
- `template<typename _Iterator, typename _Sequence >`
`bool __foreign_iterator_aux4 (const __Safe_iterator< _Iterator, _Sequence > &,...)`
- `template<typename _Iterator >`
`_Distance_traits< _Iterator >::__type __get_distance (const std::reverse_iterator< _Iterator > &__first, const std::reverse_iterator< _Iterator > &__last)`
- `template<typename _Iterator >`
`_Distance_traits< _Iterator >::__type __get_distance (const _Iterator &__lhs, const _Iterator &__rhs, std::__random_access_iterator_tag)`
- `template<typename _Iterator >`
`_Distance_traits< _Iterator >::__type __get_distance (const _Iterator &__lhs, const _Iterator &__rhs, std::__input_iterator_tag)`
- `template<typename _Iterator >`
`_Distance_traits< _Iterator >::__type __get_distance (const std::move_iterator< _Iterator > &__first, const std::move_iterator< _Iterator > &__last)`
- `template<typename _Iterator >`
`_Distance_traits< _Iterator >::__type __get_distance (const _Iterator &__lhs, const _Iterator &__rhs)`
- `template<typename _Iterator, typename _Sequence >`
`std::pair< typename std::iterator_traits< _Iterator >::difference_type, _Distance_precision > __get_distance (const __Safe_local_iterator< _Iterator, _Sequence > &__first, const __Safe_local_iterator< _Iterator, _Sequence > &__last, std::__input_iterator_tag)`
- `template<typename _Iterator, typename _Sequence >`
`_Distance_traits< _Iterator >::__type __get_distance (const __Safe_iterator< _Iterator, _Sequence > &__first, const __Safe_iterator< _Iterator, _Sequence > &__last, std::__random_access_iterator_tag)`
- `template<typename _Iterator, typename _Sequence >`
`_Distance_traits< _Iterator >::__type __get_distance (const __Safe_iterator< _Iterator, _Sequence > &__first, const __Safe_iterator< _Iterator, _Sequence > &__last, std::__input_iterator_tag)`

- `template<typename _Iterator, typename _Sequence >`
`_Distance_traits< _Iterator >::__type __get_distance_from_begin (const _Safe_iterator< _Iterator, _Sequence > &__it)`
- `template<typename _Iterator, typename _Sequence >`
`_Distance_traits< _Iterator >::__type __get_distance_to_end (const _Safe_iterator< _Iterator, _Sequence > &__it)`
- `template<typename _Iterator >`
`bool __is_irreflexive (_Iterator __it)`
- `template<typename _Iterator, typename _Pred >`
`bool __is_irreflexive_pred (_Iterator __it, _Pred __pred)`
- `template<typename _Iterator >`
`auto __unsafe (const std::reverse_iterator< _Iterator > &__it) -> decltype(std::__make_reverse_iterator(__unsafe(__it.base())))`
- `template<typename _Iterator >`
`auto __unsafe (const std::move_iterator< _Iterator > &__it) -> decltype(std::make_move_iterator(__unsafe(__it.base())))`
- `template<typename _Iterator >`
`_Iterator __unsafe (_Iterator __it)`
- `template<typename _Iterator, typename _Sequence >`
`_Iterator __unsafe (const _Safe_local_iterator< _Iterator, _Sequence > &__it)`
- `template<typename _Iterator, typename _Sequence >`
`_Iterator __unsafe (const _Safe_iterator< _Iterator, _Sequence > &__it)`
- `template<typename _Iterator >`
`bool __valid_range (const std::reverse_iterator< _Iterator > &__first, const std::reverse_iterator< _Iterator > &__last, typename _Distance_traits< _Iterator >::__type &__dist)`
- `template<typename _Iterator >`
`bool __valid_range (const std::move_iterator< _Iterator > &__first, const std::move_iterator< _Iterator > &__last, typename _Distance_traits< _Iterator >::__type &__dist)`
- `template<typename _InputIterator >`
`bool __valid_range (const _InputIterator &__first, const _InputIterator &__last, typename _Distance_traits< _InputIterator >::__type &__dist)`
- `template<typename _InputIterator >`
`bool __valid_range (const _InputIterator &__first, const _InputIterator &__last)`
- `template<typename _Iterator, typename _Sequence >`
`bool __valid_range (const _Safe_local_iterator< _Iterator, _Sequence > &__first, const _Safe_local_iterator< _Iterator, _Sequence > &__last, typename _Distance_traits< _Iterator >::__type &__dist_info)`
- `template<typename _Iterator, typename _Sequence >`
`bool __valid_range (const _Safe_iterator< _Iterator, _Sequence > &__first, const _Safe_iterator< _Iterator, _Sequence > &__last, typename _Distance_traits< _Iterator >::__type &__dist)`
- `template<typename _Integral >`
`bool __valid_range_aux (const _Integral &, const _Integral &, typename _Distance_traits< _Integral >::__type &__dist, std::__true_type)`
- `template<typename _InputIterator >`
`bool __valid_range_aux (const _InputIterator &__first, const _InputIterator &__last, typename _Distance_traits< _InputIterator >::__type &__dist, std::__false_type)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`std::basic_istream< _CharT, _Traits > &getline (std::basic_istream< _CharT, _Traits > &__is, basic_string< _CharT, _Traits, _Allocator > &__str, _CharT __delim)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`std::basic_istream< _CharT, _Traits > &getline (std::basic_istream< _CharT, _Traits > &__is, basic_string< _CharT, _Traits, _Allocator > &__str)`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`bool operator!= (const _Safe_local_iterator< _IteratorL, _Sequence > &__lhs, const _Safe_local_iterator< _IteratorR, _Sequence > &__rhs)`

- `template<typename _Iterator, typename _Sequence >`
`bool operator!= (const _Safe_local_iterator< _Iterator, _Sequence > &__lhs, const _Safe_local_iterator< _Iterator, _Sequence > &__rhs)`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`bool operator!= (const _Safe_iterator< _IteratorL, _Sequence > &__lhs, const _Safe_iterator< _IteratorR, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`bool operator!= (const _Safe_iterator< _Iterator, _Sequence > &__lhs, const _Safe_iterator< _Iterator, _Sequence > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool operator!= (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const basic_string< _CharT, _Traits, _Allocator > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool operator!= (const _CharT *__lhs, const basic_string< _CharT, _Traits, _Allocator > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool operator!= (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const _CharT *__rhs)`
- `template<typename _Iterator, typename _Sequence >`
`_Safe_iterator< _Iterator, _Sequence > operator+ (typename _Safe_iterator< _Iterator, _Sequence >::difference_type __n, const _Safe_iterator< _Iterator, _Sequence > &__i) noexcept`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`basic_string< _CharT, _Traits, _Allocator > operator+ (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const basic_string< _CharT, _Traits, _Allocator > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`basic_string< _CharT, _Traits, _Allocator > operator+ (const _CharT *__lhs, const basic_string< _CharT, _Traits, _Allocator > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`basic_string< _CharT, _Traits, _Allocator > operator+ (_CharT __lhs, const basic_string< _CharT, _Traits, _Allocator > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`basic_string< _CharT, _Traits, _Allocator > operator+ (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`basic_string< _CharT, _Traits, _Allocator > operator+ (const basic_string< _CharT, _Traits, _Allocator > &__lhs, _CharT __rhs)`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`_Safe_iterator< _IteratorL, _Sequence >::difference_type operator- (const _Safe_iterator< _IteratorL, _Sequence > &__lhs, const _Safe_iterator< _IteratorR, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`_Safe_iterator< _Iterator, _Sequence >::difference_type operator- (const _Safe_iterator< _Iterator, _Sequence > &__lhs, const _Safe_iterator< _Iterator, _Sequence > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`bool operator< (const _Safe_iterator< _IteratorL, _Sequence > &__lhs, const _Safe_iterator< _IteratorR, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`bool operator< (const _Safe_iterator< _Iterator, _Sequence > &__lhs, const _Safe_iterator< _Iterator, _Sequence > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool operator< (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const basic_string< _CharT, _Traits, _Allocator > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool operator< (const _CharT *__lhs, const basic_string< _CharT, _Traits, _Allocator > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool operator< (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const _CharT *__rhs)`

- `template<typename _CharT, typename _Traits, typename _Allocator >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`basic_string< _CharT, _Traits, _Allocator > &__str)`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`bool operator<= (const _Safe_iterator< _IteratorL, _Sequence > &__lhs, const _Safe_iterator< _IteratorR, _`
`Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`bool operator<= (const _Safe_iterator< _Iterator, _Sequence > &__lhs, const _Safe_iterator< _Iterator, _`
`Sequence > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool operator<= (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const basic_string< _CharT, _Traits,`
`_Allocator > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool operator<= (const _CharT * __lhs, const basic_string< _CharT, _Traits, _Allocator > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool operator<= (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const _CharT * __rhs)`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`bool operator== (const _Safe_local_iterator< _IteratorL, _Sequence > &__lhs, const _Safe_local_iterator< _`
`IteratorR, _Sequence > &__rhs)`
- `template<typename _Iterator, typename _Sequence >`
`bool operator== (const _Safe_local_iterator< _Iterator, _Sequence > &__lhs, const _Safe_local_iterator< _`
`Iterator, _Sequence > &__rhs)`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`bool operator== (const _Safe_iterator< _IteratorL, _Sequence > &__lhs, const _Safe_iterator< _IteratorR, _`
`Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`bool operator== (const _Safe_iterator< _Iterator, _Sequence > &__lhs, const _Safe_iterator< _Iterator, _`
`Sequence > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool operator== (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const basic_string< _CharT, _Traits,`
`_Allocator > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool operator== (const _CharT * __lhs, const basic_string< _CharT, _Traits, _Allocator > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool operator== (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const _CharT * __rhs)`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`bool operator> (const _Safe_iterator< _IteratorL, _Sequence > &__lhs, const _Safe_iterator< _IteratorR, _`
`Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`bool operator> (const _Safe_iterator< _Iterator, _Sequence > &__lhs, const _Safe_iterator< _Iterator, _`
`Sequence > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool operator> (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const basic_string< _CharT, _Traits,`
`_Allocator > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool operator> (const _CharT * __lhs, const basic_string< _CharT, _Traits, _Allocator > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool operator> (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const _CharT * __rhs)`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`bool operator>= (const _Safe_iterator< _IteratorL, _Sequence > &__lhs, const _Safe_iterator< _IteratorR, _`
`Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`bool operator>= (const _Safe_iterator< _Iterator, _Sequence > &__lhs, const _Safe_iterator< _Iterator, _`
`Sequence > &__rhs) noexcept`

- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool operator>= (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const basic_string< _CharT, _Traits, _Allocator > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool operator>= (const _CharT * __lhs, const basic_string< _CharT, _Traits, _Allocator > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`bool operator>= (const basic_string< _CharT, _Traits, _Allocator > &__lhs, const _CharT * __rhs)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, basic_istream< _CharT, _Traits, _Allocator > &__str)`
- `template<typename _CharT, typename _Traits, typename _Allocator >`
`void swap (basic_string< _CharT, _Traits, _Allocator > &__lhs, basic_string< _CharT, _Traits, _Allocator > &__rhs)`

4.4.1 Detailed Description

GNU debug classes for public use.

4.4.2 Enumeration Type Documentation

4.4.2.1 `enum __gnu_debug::Distance_precision`

The precision to which we can calculate the distance between two iterators.

Definition at line 43 of file `helper_functions.h`.

4.4.3 Function Documentation

4.4.3.1 `template<typename _Iterator > bool __gnu_debug::__check_dereferenceable (const _Iterator &) [inline]`

Assume that some arbitrary iterator is dereferenceable, because we can't prove that it isn't.

Definition at line 75 of file `functions.h`.

Referenced by `__gnu_debug::Safe_iterator< _Iterator, _Sequence >::M_incrementable()`.

4.4.3.2 `template<typename _Tp > bool __gnu_debug::__check_dereferenceable (const _Tp * __ptr) [inline]`

Non-NULL pointers are dereferenceable.

Definition at line 81 of file `functions.h`.

References `std::__addressof()`.

4.4.3.3 `template<typename _Iterator, typename _Sequence> bool __gnu_debug::__check_dereferenceable (const
_Safe_local_iterator<_Iterator, _Sequence> &__x) [inline]`

Safe local iterators know if they are dereferenceable.

Definition at line 437 of file `safe_local_iterator.h`.

4.4.3.4 `template<typename _Iterator, typename _Sequence> bool __gnu_debug::__check_dereferenceable (const
_Safe_iterator<_Iterator, _Sequence> &__x) [inline]`

Safe iterators know if they are dereferenceable.

Definition at line 744 of file `safe_iterator.h`.

References `__gnu_debug::_Safe_iterator<_Iterator, _Sequence>::_M_dereferenceable()`.

4.4.3.5 `template<typename _Tp> bool __gnu_debug::__check_singular (const _Tp* __ptr) [inline]`

Non-NULL pointers are nonsingular.

Definition at line 68 of file `functions.h`.

4.4.3.6 `bool __gnu_debug::__check_singular_aux (const _Safe_iterator_base* __x) [inline]`

Iterators that derive from `_Safe_iterator_base` can be determined singular or non-singular.

Definition at line 166 of file `safe_base.h`.

References `__gnu_debug::_Safe_iterator_base::_M_singular()`.

4.4.3.7 `template<typename _CharT, typename _Integer> const _CharT* __gnu_debug::__check_string (const _CharT* __s,
const _Integer &__n __attribute__((unused))) [inline]`

Checks that `__s` is non-NULL or `__n == 0`, and then returns `__s`.

Definition at line 218 of file `functions.h`.

4.4.3.8 `template<typename _CharT> const _CharT* __gnu_debug::__check_string (const _CharT* __s) [inline]`

Checks that `__s` is non-NULL and then returns `__s`.

Definition at line 230 of file `functions.h`.

References `std::_iterator_category()`.

```
4.4.3.9  template<typename _Iterator, typename _Sequence, typename _OtherIterator> bool __gnu_debug::__foreign_iterator_aux2
        ( const _Safe_iterator< _Iterator, _Sequence> & __it, const _Safe_iterator< _OtherIterator, _Sequence> & __other,
          const _Safe_iterator< _OtherIterator, _Sequence> & ) [inline]
```

Handle debug iterators from the same type of container.

Definition at line 151 of file functions.h.

Referenced by __foreign_iterator_aux2().

```
4.4.3.10 template<typename _Iterator, typename _Sequence, typename _OtherIterator, typename _OtherSequence> bool
        __gnu_debug::__foreign_iterator_aux2( const _Safe_iterator< _Iterator, _Sequence> & __it, const _Safe_iterator<
        _OtherIterator, _OtherSequence> &, const _Safe_iterator< _OtherIterator, _OtherSequence> & ) [inline]
```

Handle debug iterators from different types of container.

Definition at line 160 of file functions.h.

References __foreign_iterator_aux2().

```
4.4.3.11 template<typename _Iterator> _Distance_traits<_Iterator>::__type __gnu_debug::__get_distance( const _Iterator &
        __lhs, const _Iterator & __rhs, std::random_access_iterator_tag ) [inline]
```

Determine the distance between two iterators with some known precision.

Definition at line 83 of file helper_functions.h.

References std::__iterator_category(), and std::make_pair().

Referenced by __get_distance(), and __valid_range_aux().

```
4.4.3.12 template<typename _Iterator, typename _Sequence> std::pair<typename std::iterator_traits<_Iterator>::difference_type,
        _Distance_precision> __gnu_debug::__get_distance( const _Safe_local_iterator< _Iterator, _Sequence>
        & __first, const _Safe_local_iterator< _Iterator, _Sequence> & __last, std::input_iterator_tag ) [inline]
```

Safe local iterators need a special method to get distance between each other.

Definition at line 454 of file safe_local_iterator.h.

```
4.4.3.13 template<typename _Iterator, typename _Sequence> _Distance_traits<_Iterator>::__type __gnu_debug::__get_distance
        ( const _Safe_iterator< _Iterator, _Sequence> & __first, const _Safe_iterator< _Iterator, _Sequence> & __last,
          std::random_access_iterator_tag ) [inline]
```

Safe iterators can help to get better distance knowledge.

Definition at line 758 of file safe_iterator.h.

References __get_distance(), std::__iterator_category(), __gnu_debug::__Safe_iterator< _Iterator, _Sequence>::__M_is_before_begin(), __gnu_debug::__Safe_iterator< _Iterator, _Sequence>::__M_is_begin(), __gnu_debug::__Safe_iterator< _Iterator, _Sequence>::__M_is_end(), __gnu_debug::__Safe_iterator< _Iterator, _Sequence>::__base(), std::make_pair(), and std::pair< _T1, _T2>::__second.


```
4.4.3.14 template<typename _InputIterator > bool __gnu_debug::__valid_range ( const _InputIterator & __first, const _InputIterator
& __last, typename _Distance_traits< _InputIterator >::__type & __dist ) [inline]
```

Don't know what these iterators are, or if they are even iterators (we may get an integral type for `_InputIterator`), so see if they are integral and pass them on to the next phase otherwise.

Definition at line 151 of file `helper_functions.h`.

References `__valid_range_aux()`.

```
4.4.3.15 template<typename _Iterator , typename _Sequence > bool __gnu_debug::__valid_range ( const
_Safe_local_iterator< _Iterator, _Sequence > & __first, const _Safe_local_iterator< _Iterator, _Sequence > &
__last, typename _Distance_traits< _Iterator >::__type & __dist_info ) [inline]
```

Safe local iterators know how to check if they form a valid range.

Definition at line 444 of file `safe_local_iterator.h`.

```
4.4.3.16 template<typename _Iterator , typename _Sequence > bool __gnu_debug::__valid_range ( const _Safe_iterator<
_Iterator, _Sequence > & __first, const _Safe_iterator< _Iterator, _Sequence > & __last, typename _Distance_traits<
_Iterator >::__type & __dist ) [inline]
```

Safe iterators know how to check if they form a valid range.

Definition at line 750 of file `safe_iterator.h`.

```
4.4.3.17 template<typename _Integral > bool __gnu_debug::__valid_range_aux ( const _Integral & , const _Integral & , typename
_Distance_traits< _Integral >::__type & __dist, std::__true_type ) [inline]
```

We say that integral types for a valid range, and defer to other routines to realize what to do with integral types instead of iterators.

Definition at line 109 of file `helper_functions.h`.

References `std::make_pair()`.

Referenced by `__valid_range()`.

```
4.4.3.18 template<typename _InputIterator > bool __gnu_debug::__valid_range_aux ( const _InputIterator & __first, const
_InputIterator & __last, typename _Distance_traits< _InputIterator >::__type & __dist, std::__false_type ) [inline]
```

We have iterators, so figure out what kind of iterators that are to see if we can check the range ahead of time.

Definition at line 122 of file `helper_functions.h`.

References `__get_distance()`, `std::pair< _T1, _T2 >::first`, and `std::pair< _T1, _T2 >::second`.

4.5 `__gnu_internal` Namespace Reference

4.5.1 Detailed Description

GNU implementation details, not for public use or export. Used only when anonymous namespaces cannot be substituted.

4.6 `__gnu_parallel` Namespace Reference

Classes

- struct [__accumulate_binop_reduct](#)
- struct [__accumulate_selector](#)
- struct [__adjacent_difference_selector](#)
- struct [__adjacent_find_selector](#)
- class [__binder1st](#)
- class [__binder2nd](#)
- struct [__count_if_selector](#)
- struct [__count_selector](#)
- struct [__fill_selector](#)
- struct [__find_first_of_selector](#)
- struct [__find_if_selector](#)
- struct [__for_each_selector](#)
- struct [__generate_selector](#)
- struct [__generic_find_selector](#)
- struct [__generic_for_each_selector](#)
- struct [__identity_selector](#)
- struct [__inner_product_selector](#)
- struct [__max_element_reduct](#)
- struct [__min_element_reduct](#)
- struct [__mismatch_selector](#)
- struct [__multiway_merge_3_variant_sentinel_switch](#)
- struct [__multiway_merge_3_variant_sentinel_switch< true, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >](#)
- struct [__multiway_merge_4_variant_sentinel_switch](#)
- struct [__multiway_merge_4_variant_sentinel_switch< true, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >](#)
- struct [__multiway_merge_k_variant_sentinel_switch](#)
- struct [__multiway_merge_k_variant_sentinel_switch< false, __stable, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >](#)
- struct [__replace_if_selector](#)
- struct [__replace_selector](#)
- struct [__transform1_selector](#)
- struct [__transform2_selector](#)
- class [__unary_negate](#)
- struct [_DRandomShufflingGlobalData](#)
- struct [_DRSSorterPU](#)
- struct [_DummyReduct](#)
- class [_EqualFromLess](#)

- struct [_EqualTo](#)
- class [_GuardedIterator](#)
- class [_IteratorPair](#)
- class [_IteratorTriple](#)
- struct [_Job](#)
- struct [_Less](#)
- class [_Lexicographic](#)
- class [_LexicographicReverse](#)
- class [_LoserTree](#)
- class [_LoserTree< false, _Tp, _Compare >](#)
- class [_LoserTreeBase](#)
- class [_LoserTreePointer](#)
- class [_LoserTreePointer< false, _Tp, _Compare >](#)
- class [_LoserTreePointerBase](#)
- class [_LoserTreePointerUnguarded](#)
- class [_LoserTreePointerUnguarded< false, _Tp, _Compare >](#)
- class [_LoserTreePointerUnguardedBase](#)
- struct [_LoserTreeTraits](#)
- class [_LoserTreeUnguarded](#)
- class [_LoserTreeUnguarded< false, _Tp, _Compare >](#)
- class [_LoserTreeUnguardedBase](#)
- struct [_Multiplies](#)
- struct [_Nothing](#)
- struct [_Piece](#)
- struct [_Plus](#)
- struct [_PMWMSSortingData](#)
- class [_PseudoSequence](#)
- class [_PseudoSequenceIterator](#)
- struct [_QSBThreadLocal](#)
- class [_RandomNumber](#)
- class [_RestrictedBoundedConcurrentQueue](#)
- struct [_SamplingSorter](#)
- struct [_SamplingSorter< false, _RAIter, _StrictWeakOrdering >](#)
- struct [_Settings](#)
- struct [_SplitConsistently](#)
- struct [_SplitConsistently< false, _RAIter, _Compare, _SortingPlacesIterator >](#)
- struct [_SplitConsistently< true, _RAIter, _Compare, _SortingPlacesIterator >](#)
- struct [balanced_quicksort_tag](#)
- struct [balanced_tag](#)
- struct [constant_size_blocks_tag](#)
- struct [default_parallel_tag](#)
- struct [equal_split_tag](#)
- struct [exact_tag](#)
- struct [find_tag](#)
- struct [growing_blocks_tag](#)
- struct [multiway_mergesort_exact_tag](#)
- struct [multiway_mergesort_sampling_tag](#)
- struct [multiway_mergesort_tag](#)
- struct [omp_loop_static_tag](#)
- struct [omp_loop_tag](#)
- struct [parallel_tag](#)

- struct [quicksort_tag](#)
- struct [sampling_tag](#)
- struct [sequential_tag](#)
- struct [unbalanced_tag](#)

Typedefs

- typedef unsigned short [_BinIndex](#)
- typedef int64_t [_CASable](#)
- typedef uint64_t [_SequenceIndex](#)
- typedef uint16_t [_ThreadIndex](#)

Enumerations

- enum [_AlgorithmStrategy](#) { **heuristic**, **force_sequential**, **force_parallel** }
- enum [_FindAlgorithm](#) { **GROWING_BLOCKS**, **CONSTANT_SIZE_BLOCKS**, **EQUAL_SPLIT** }
- enum [_MultiwayMergeAlgorithm](#) { **LOSER_TREE** }
- enum [_Parallelism](#) { [sequential](#), [parallel_unbalanced](#), [parallel_balanced](#), [parallel_omp_loop](#), [parallel_omp_loop_static](#), [parallel_taskqueue](#) }
- enum [_PartialSumAlgorithm](#) { **RECURSIVE**, **LINEAR** }
- enum [_SortAlgorithm](#) { **MWMS**, **QS**, **QS_BALANCED** }
- enum [_SplittingAlgorithm](#) { **SAMPLING**, **EXACT** }

Functions

- template<typename _Tp >
_Tp **add_omp** (volatile _Tp *__ptr, _Tp __addend)
- template<typename _RAIter, typename _DifferenceTp >
void [calc_borders](#) (_RAIter __elements, _DifferenceTp __length, _DifferenceTp *__off)
- template<typename _Tp >
bool **cas_omp** (volatile _Tp *__ptr, _Tp __comparand, _Tp __replacement)
- template<typename _Tp >
bool [compare_and_swap](#) (volatile _Tp *__ptr, _Tp __comparand, _Tp __replacement)
- template<typename _Iter, typename _OutputIterator >
_OutputIterator **copy_tail** (std::pair<_Iter, _Iter> __b, std::pair<_Iter, _Iter> __e, _OutputIterator __r)
- void [decode2](#) (_CASable __x, int &__a, int &__b)
- template<typename _RAIter, typename _DifferenceTp >
void [determine_samples](#) (_PMWMSortingData<_RAIter> *__sd, _DifferenceTp __num_samples)
- [CASable](#) [encode2](#) (int __a, int __b)
- template<typename _DifferenceType, typename _OutputIterator >
_OutputIterator [equally_split](#) (_DifferenceType __n, [ThreadIndex](#) __num_threads, _OutputIterator __s)
- template<typename _DifferenceType >
_DifferenceType [equally_split_point](#) (_DifferenceType __n, [ThreadIndex](#) __num_threads, [ThreadIndex](#) __thread_no)
- template<typename _Tp >
_Tp [fetch_and_add](#) (volatile _Tp *__ptr, _Tp __addend)

- `template<typename _RAIter1, typename _RAIter2, typename _Pred, typename _Selector >`
`std::pair< _RAIter1, _RAIter2 > __find_template (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, _Pred __pred, _Selector __selector)`
- `template<typename _RAIter1, typename _RAIter2, typename _Pred, typename _Selector >`
`std::pair< _RAIter1, _RAIter2 > __find_template (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, _Pred __pred, _Selector __selector, equal_split_tag)`
- `template<typename _RAIter1, typename _RAIter2, typename _Pred, typename _Selector >`
`std::pair< _RAIter1, _RAIter2 > __find_template (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, _Pred __pred, _Selector __selector, growing_blocks_tag)`
- `template<typename _RAIter1, typename _RAIter2, typename _Pred, typename _Selector >`
`std::pair< _RAIter1, _RAIter2 > __find_template (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, _Pred __pred, _Selector __selector, constant_size_blocks_tag)`
- `template<typename _Iter, typename _UserOp, typename _Functionality, typename _Red, typename _Result >`
`_UserOp __for_each_template_random_access (_Iter __begin, _Iter __end, _UserOp __user_op, _Functionality & __functionality, _Red __reduction, _Result __reduction_start, _Result & __output, typename std::iterator_traits< _Iter >::difference_type __bound, _Parallelism __parallelism_tag)`
- `template<typename _RAIter, typename _Op, typename _Fu, typename _Red, typename _Result >`
`_Op __for_each_template_random_access_ed (_RAIter __begin, _RAIter __end, _Op __o, _Fu & __f, _Red __r, _Result __base, _Result & __output, typename std::iterator_traits< _RAIter >::difference_type __bound)`
- `template<typename _RAIter, typename _Op, typename _Fu, typename _Red, typename _Result >`
`_Op __for_each_template_random_access_omp_loop (_RAIter __begin, _RAIter __end, _Op __o, _Fu & __f, _Red __r, _Result __base, _Result & __output, typename std::iterator_traits< _RAIter >::difference_type __bound)`
- `template<typename _RAIter, typename _Op, typename _Fu, typename _Red, typename _Result >`
`_Op __for_each_template_random_access_omp_loop_static (_RAIter __begin, _RAIter __end, _Op __o, _Fu & __f, _Red __r, _Result __base, _Result & __output, typename std::iterator_traits< _RAIter >::difference_type __bound)`
- `template<typename _RAIter, typename _Op, typename _Fu, typename _Red, typename _Result >`
`_Op __for_each_template_random_access_workstealing (_RAIter __begin, _RAIter __end, _Op __op, _Fu & __f, _Red __r, _Result __base, _Result & __output, typename std::iterator_traits< _RAIter >::difference_type __bound)`
- `__ThreadIndex __get_max_threads ()`
- `bool __is_parallel (const _Parallelism __p)`
- `template<typename _Iter, typename _Compare >`
`bool __is_sorted (_Iter __begin, _Iter __end, _Compare __comp)`
- `template<typename _RAIter, typename _Compare >`
`_RAIter __median_of_three_iterators (_RAIter __a, _RAIter __b, _RAIter __c, _Compare __comp)`
- `template<typename _RAIter1, typename _RAIter2, typename _OutputIterator, typename _DifferenceTp, typename _Compare >`
`_OutputIterator __merge_advance (_RAIter1 & __begin1, _RAIter1 __end1, _RAIter2 & __begin2, _RAIter2 __end2, _OutputIterator __target, _DifferenceTp __max_length, _Compare __comp)`
- `template<typename _RAIter1, typename _RAIter2, typename _OutputIterator, typename _DifferenceTp, typename _Compare >`
`_OutputIterator __merge_advance_movc (_RAIter1 & __begin1, _RAIter1 __end1, _RAIter2 & __begin2, _RAIter2 __end2, _OutputIterator __target, _DifferenceTp __max_length, _Compare __comp)`
- `template<typename _RAIter1, typename _RAIter2, typename _OutputIterator, typename _DifferenceTp, typename _Compare >`
`_OutputIterator __merge_advance_usual (_RAIter1 & __begin1, _RAIter1 __end1, _RAIter2 & __begin2, _RAIter2 __end2, _OutputIterator __target, _DifferenceTp __max_length, _Compare __comp)`
- `template<typename _RAIter1, typename _RAIter2, typename _RAIter3, typename _Compare >`
`_RAIter3 __parallel_merge_advance (_RAIter1 & __begin1, _RAIter1 __end1, _RAIter2 & __begin2, _RAIter2 __end2, _RAIter3 __target, typename std::iterator_traits< _RAIter1 >::difference_type __max_length, _Compare __comp)`
- `template<typename _RAIter1, typename _RAIter3, typename _Compare >`
`_RAIter3 __parallel_merge_advance (_RAIter1 & __begin1, _RAIter1 __end1, _RAIter1 & __begin2, _RAIter1 __end2, _RAIter3 __target, typename std::iterator_traits< _RAIter1 >::difference_type __max_length, _Compare __comp)`

- `template<typename _RAIter, typename _Compare >`
`void __parallel_nth_element (_RAIter __begin, _RAIter __nth, _RAIter __end, _Compare __comp)`
- `template<typename _RAIter, typename _Compare >`
`void __parallel_partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end, _Compare __comp)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator __parallel_partial_sum (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __bin_op)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator __parallel_partial_sum_basecase (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __bin_op, typename std::iterator_traits< _Iter >::value_type __value)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator __parallel_partial_sum_linear (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __bin_op, typename std::iterator_traits< _Iter >::difference_type __n)`
- `template<typename _RAIter, typename _Predicate >`
`std::iterator_traits< _RAIter >::difference_type __parallel_partition (_RAIter __begin, _RAIter __end, _Predicate __pred, _ThreadIndex __num_threads)`
- `template<typename _RAIter, typename _RandomNumberGenerator >`
`void __parallel_random_shuffle (_RAIter __begin, _RAIter __end, _RandomNumberGenerator __rng= RandomNumber())`
- `template<typename _RAIter, typename _RandomNumberGenerator >`
`void __parallel_random_shuffle_drs (_RAIter __begin, _RAIter __end, typename std::iterator_traits< _RAIter >::difference_type __n, _ThreadIndex __num_threads, _RandomNumberGenerator &__rng)`
- `template<typename _RAIter, typename _RandomNumberGenerator >`
`void __parallel_random_shuffle_drs_pu (DRSSorterPU< _RAIter, _RandomNumberGenerator > *__pus)`
- `template<typename _Iter, typename _OutputIterator, typename _Compare >`
`_OutputIterator __parallel_set_difference (_Iter __begin1, _Iter __end1, _Iter __begin2, _Iter __end2, _OutputIterator __result, _Compare __comp)`
- `template<typename _Iter, typename _OutputIterator, typename _Compare >`
`_OutputIterator __parallel_set_intersection (_Iter __begin1, _Iter __end1, _Iter __begin2, _Iter __end2, _OutputIterator __result, _Compare __comp)`
- `template<typename _Iter, typename _OutputIterator, typename _Operation >`
`_OutputIterator __parallel_set_operation (_Iter __begin1, _Iter __end1, _Iter __begin2, _Iter __end2, _OutputIterator __result, _Operation __op)`
- `template<typename _Iter, typename _OutputIterator, typename _Compare >`
`_OutputIterator __parallel_set_symmetric_difference (_Iter __begin1, _Iter __end1, _Iter __begin2, _Iter __end2, _OutputIterator __result, _Compare __comp)`
- `template<typename _Iter, typename _OutputIterator, typename _Compare >`
`_OutputIterator __parallel_set_union (_Iter __begin1, _Iter __end1, _Iter __begin2, _Iter __end2, _OutputIterator __result, _Compare __comp)`
- `template<bool __stable, typename _RAIter, typename _Compare, typename _Parallelism >`
`void __parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, _Parallelism __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, multiway_mergesort_tag __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, multiway_mergesort_exact_tag __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, multiway_mergesort_sampling_tag __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, quicksort_tag __parallelism)`

- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, balanced_quicksort_tag __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, default_parallel_tag __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, parallel_tag __parallelism)`
- `template<typename _RAIter, typename _Compare >`
`void __parallel_sort_qs (_RAIter __begin, _RAIter __end, _Compare __comp, ThreadIndex __num_threads)`
- `template<typename _RAIter, typename _Compare >`
`void __parallel_sort_qs_conquer (_RAIter __begin, _RAIter __end, _Compare __comp, ThreadIndex __num_threads)`
- `template<typename _RAIter, typename _Compare >`
`std::iterator_traits< _RAIter >::difference_type __parallel_sort_qs_divide (_RAIter __begin, _RAIter __end, _Compare __comp, typename std::iterator_traits< _RAIter >::difference_type __pivot_rank, typename std::iterator_traits< _RAIter >::difference_type __num_samples, ThreadIndex __num_threads)`
- `template<typename _RAIter, typename _Compare >`
`void __parallel_sort_qsb (_RAIter __begin, _RAIter __end, _Compare __comp, ThreadIndex __num_threads)`
- `template<typename _Iter, class _OutputIterator, class _BinaryPredicate >`
`_OutputIterator __parallel_unique_copy (_Iter __first, _Iter __last, _OutputIterator __result, _BinaryPredicate __binary_pred)`
- `template<typename _Iter, class _OutputIterator >`
`_OutputIterator __parallel_unique_copy (_Iter __first, _Iter __last, _OutputIterator __result)`
- `template<typename _RAIter, typename _Compare >`
`void __qsb_conquer (QSBThreadLocal< _RAIter > ** __tls, _RAIter __begin, _RAIter __end, _Compare __comp, ThreadIndex __iam, ThreadIndex __num_threads, bool __parent_wait)`
- `template<typename _RAIter, typename _Compare >`
`std::iterator_traits< _RAIter >::difference_type __qsb_divide (_RAIter __begin, _RAIter __end, _Compare __comp, ThreadIndex __num_threads)`
- `template<typename _RAIter, typename _Compare >`
`void __qsb_local_sort_with_helping (QSBThreadLocal< _RAIter > ** __tls, _Compare & __comp, ThreadIndex __iam, bool __wait)`
- `template<typename _RandomNumberGenerator >`
`int __random_number_pow2 (int __logp, _RandomNumberGenerator & __rng)`
- `template<typename _Size >`
`_Size __rd_log2 (_Size __n)`
- `template<typename _Tp >`
`_Tp __round_up_to_pow2 (_Tp __x)`
- `template<typename __RAIter1, typename __RAIter2, typename _Pred >`
`__RAIter1 __search_template (__RAIter1 __begin1, __RAIter1 __end1, __RAIter2 __begin2, __RAIter2 __end2, _Pred __pred)`
- `template<bool __stable, bool __sentinels, typename _RAIterIterator, typename _RAIter3, typename _DifferenceTp, typename _Compare >`
`_RAIter3 __sequential_multiway_merge (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 __target, const typename std::iterator_traits< typename std::iterator_traits< _RAIterIterator >::value_type >::first_type >::value_type & __sentinel, _DifferenceTp __length, _Compare __comp)`
- `template<typename _RAIter, typename _RandomNumberGenerator >`
`void __sequential_random_shuffle (_RAIter __begin, _RAIter __end, _RandomNumberGenerator & __rng)`
- `template<typename _Iter >`
`void __shrink (std::vector< _Iter > & __os_starts, size_t & __count_to_two, size_t & __range_length)`
- `template<typename _Iter >`
`void __shrink_and_double (std::vector< _Iter > & __os_starts, size_t & __count_to_two, size_t & __range_length, const bool __make_twice)`
- `void __yield ()`

- `template<typename _Iter, typename _FuncType >`
`size_t list_partition (const _Iter __begin, const _Iter __end, _Iter *__starts, size_t *__lengths, const int __↵`
`num_parts, _FuncType &__f, int __oversampling=0)`
- `template<typename _Tp >`
`const _Tp &max (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp >`
`const _Tp &min (const _Tp &__a, const _Tp &__b)`
- `template<typename _RanSeqs, typename _RankType, typename _RankIterator, typename _Compare >`
`void multiseq_partition (_RanSeqs __begin_seqs, _RanSeqs __end_seqs, _RankType __rank, _RankIterator __↵`
`__begin_offsets, _Compare __comp=std::less< typename std::iterator_traits< typename std::iterator_traits< ↵`
`_RanSeqs >::value_type::first_type >::value_type >())`
- `template<typename _Tp, typename _RanSeqs, typename _RankType, typename _Compare >`
`_Tp multiseq_selection (_RanSeqs __begin_seqs, _RanSeqs __end_seqs, _RankType __rank, _RankType &__↵`
`__offset, _Compare __comp=std::less< _Tp >())`
- `template<typename _RAlterPairIterator, typename _RAlterOut, typename _DifferenceTp, typename _Compare >`
`_RAlterOut multiway_merge (_RAlterPairIterator __seqs_begin, _RAlterPairIterator __seqs_end, _RAlterOut __↵`
`__target, _DifferenceTp __length, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _RAlterPairIterator, typename _RAlterOut, typename _DifferenceTp, typename _Compare >`
`_RAlterOut multiway_merge (_RAlterPairIterator __seqs_begin, _RAlterPairIterator __seqs_end, _RAlterOut __↵`
`__target, _DifferenceTp __length, _Compare __comp, __gnu_parallel::exact_tag __tag)`
- `template<typename _RAlterPairIterator, typename _RAlterOut, typename _DifferenceTp, typename _Compare >`
`_RAlterOut multiway_merge (_RAlterPairIterator __seqs_begin, _RAlterPairIterator __seqs_end, _RAlterOut __↵`
`__target, _DifferenceTp __length, _Compare __comp, __gnu_parallel::sampling_tag __tag)`
- `template<typename _RAlterPairIterator, typename _RAlterOut, typename _DifferenceTp, typename _Compare >`
`_RAlterOut multiway_merge (_RAlterPairIterator __seqs_begin, _RAlterPairIterator __seqs_end, _RAlterOut __↵`
`__target, _DifferenceTp __length, _Compare __comp, parallel_tag __tag=parallel_tag(0))`
- `template<typename _RAlterPairIterator, typename _RAlterOut, typename _DifferenceTp, typename _Compare >`
`_RAlterOut multiway_merge (_RAlterPairIterator __seqs_begin, _RAlterPairIterator __seqs_end, _RAlterOut __↵`
`__target, _DifferenceTp __length, _Compare __comp, default_parallel_tag __tag)`
- `template<template< typename RAI, typename C > class iterator, typename _RAlterIterator, typename _RAlter3, typename _DifferenceTp`
`, typename _Compare >`
`_RAlter3 multiway_merge_3_variant (_RAlterIterator __seqs_begin, _RAlterIterator __seqs_end, _RAlter3 __↵`
`target, _DifferenceTp __length, _Compare __comp)`
- `template<template< typename RAI, typename C > class iterator, typename _RAlterIterator, typename _RAlter3, typename _DifferenceTp`
`, typename _Compare >`
`_RAlter3 multiway_merge_4_variant (_RAlterIterator __seqs_begin, _RAlterIterator __seqs_end, _RAlter3 __↵`
`target, _DifferenceTp __length, _Compare __comp)`
- `template<bool __stable, typename _RAlterIterator, typename _Compare, typename _DifferenceType >`
`void multiway_merge_exact_splitting (_RAlterIterator __seqs_begin, _RAlterIterator __seqs_end, _Difference↵`
`Type __length, _DifferenceType __total_length, _Compare __comp, std::vector< std::pair< _DifferenceType, ↵`
`_DifferenceType > > *__pieces)`
- `template<typename _LT, typename _RAlterIterator, typename _RAlter3, typename _DifferenceTp, typename _Compare >`
`_RAlter3 multiway_merge_loser_tree (_RAlterIterator __seqs_begin, _RAlterIterator __seqs_end, _RAlter3 __↵`
`__target, _DifferenceTp __length, _Compare __comp)`
- `template<typename UnguardedLoserTree, typename _RAlterIterator, typename _RAlter3, typename _DifferenceTp, typename _Compare`
`>`
`_RAlter3 multiway_merge_loser_tree_sentinel (_RAlterIterator __seqs_begin, _RAlterIterator __seqs_end, _R↵`
`Alter3 __target, const typename std::iterator_traits< typename std::iterator_traits< _RAlterIterator >::value↵`
`type::first_type >::value_type &__sentinel, _DifferenceTp __length, _Compare __comp)`
- `template<typename _LT, typename _RAlterIterator, typename _RAlter3, typename _DifferenceTp, typename _Compare >`
`_RAlter3 multiway_merge_loser_tree_unguarded (_RAlterIterator __seqs_begin, _RAlterIterator __seqs_end, ↵`
`_RAlter3 __target, const typename std::iterator_traits< typename std::iterator_traits< _RAlterIterator >::value↵`
`_type::first_type >::value_type &__sentinel, _DifferenceTp __length, _Compare __comp)`

- `template<bool __stable, typename _RAIterlterator, typename _Compare, typename _DifferenceType >`
`void multiway_merge_sampling_splitting (_RAIterlterator __seqs_begin, _RAIterlterator __seqs_end, ↵`
`_DifferenceType __length, _DifferenceType __total_length, _Compare __comp, std::vector< std::pair< ↵`
`_DifferenceType, _DifferenceType > > *__pieces)`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut multiway_merge_sentinels (_RAIterPairlterator __seqs_begin, _RAIterPairlterator __seqs_end, _R↵`
`_RAIterOut __target, _DifferenceTp __length, _Compare __comp, gnu_parallel::sequential_tag)`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut multiway_merge_sentinels (_RAIterPairlterator __seqs_begin, _RAIterPairlterator __seqs_end, ↵`
`_RAIterOut __target, _DifferenceTp __length, _Compare __comp, gnu_parallel::exact_tag __tag)`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut multiway_merge_sentinels (_RAIterPairlterator __seqs_begin, _RAIterPairlterator __seqs_end, ↵`
`_RAIterOut __target, _DifferenceTp __length, _Compare __comp, sampling_tag __tag)`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut multiway_merge_sentinels (_RAIterPairlterator __seqs_begin, _RAIterPairlterator __seqs_end, ↵`
`_RAIterOut __target, _DifferenceTp __length, _Compare __comp, parallel_tag __tag=parallel_tag(0))`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut multiway_merge_sentinels (_RAIterPairlterator __seqs_begin, _RAIterPairlterator __seqs_end, ↵`
`_RAIterOut __target, _DifferenceTp __length, _Compare __comp, default_parallel_tag __tag)`
- `template<bool __stable, bool __sentinels, typename _RAIterlterator, typename _RAIter3, typename _DifferenceTp, typename _Splitter,`
`typename _Compare >`
`_RAIter3 parallel_multiway_merge (_RAIterlterator __seqs_begin, _RAIterlterator __seqs_end, _RAIter3 __↵`
`target, _Splitter __splitter, _DifferenceTp __length, _Compare __comp, ThreadIndex __num_threads)`
- `template<bool __stable, bool __exact, typename _RAIter, typename _Compare >`
`void parallel_sort_mwms (_RAIter __begin, _RAIter __end, _Compare __comp, ThreadIndex __num_threads)`
- `template<bool __stable, bool __exact, typename _RAIter, typename _Compare >`
`void parallel_sort_mwms_pu (PMWMSortingData< _RAIter > * __sd, _Compare & __comp)`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut stable_multiway_merge (_RAIterPairlterator __seqs_begin, _RAIterPairlterator __seqs_end, _R↵`
`_RAIterOut __target, _DifferenceTp __length, _Compare __comp, gnu_parallel::sequential_tag)`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut stable_multiway_merge (_RAIterPairlterator __seqs_begin, _RAIterPairlterator __seqs_end, _R↵`
`_RAIterOut __target, _DifferenceTp __length, _Compare __comp, gnu_parallel::exact_tag __tag)`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut stable_multiway_merge (_RAIterPairlterator __seqs_begin, _RAIterPairlterator __seqs_end, _R↵`
`_RAIterOut __target, _DifferenceTp __length, _Compare __comp, sampling_tag __tag)`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut stable_multiway_merge (_RAIterPairlterator __seqs_begin, _RAIterPairlterator __seqs_end, _R↵`
`_RAIterOut __target, _DifferenceTp __length, _Compare __comp, parallel_tag __tag=parallel_tag(0))`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut stable_multiway_merge (_RAIterPairlterator __seqs_begin, _RAIterPairlterator __seqs_end, _R↵`
`_RAIterOut __target, _DifferenceTp __length, _Compare __comp, default_parallel_tag __tag)`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut stable_multiway_merge_sentinels (_RAIterPairlterator __seqs_begin, _RAIterPairlterator ↵`
`seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, gnu_parallel::sequential_tag)`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut stable_multiway_merge_sentinels (_RAIterPairlterator __seqs_begin, _RAIterPairlterator ↵`
`seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, gnu_parallel::exact_tag __tag)`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut stable_multiway_merge_sentinels (_RAIterPairlterator __seqs_begin, _RAIterPairlterator ↵`
`seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, sampling_tag __tag)`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut stable_multiway_merge_sentinels (_RAIterPairlterator __seqs_begin, _RAIterPairlterator ↵`
`seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, parallel_tag __tag=parallel_tag(0))`

- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut stable_multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __↔`
`seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, default_parallel_tag __tag)`

Variables

- `static const int _CASable_bits`
- `static const _CASable _CASable_mask`

4.6.1 Detailed Description

GNU parallel code for public use.

4.6.2 Typedef Documentation

4.6.2.1 `typedef unsigned short __gnu_parallel::_BinIndex`

Type to hold the index of a bin.

Since many variables of this type are allocated, it should be chosen as small as possible.

Definition at line 47 of file `random_shuffle.h`.

4.6.2.2 `typedef int64_t __gnu_parallel::_CASable`

Longest compare-and-swappable integer type on this platform.

Definition at line 127 of file `types.h`.

4.6.2.3 `typedef uint64_t __gnu_parallel::_SequenceIndex`

Unsigned integer to index `__elements`. The total number of elements for each algorithm must fit into this type.

Definition at line 117 of file `types.h`.

4.6.2.4 `typedef uint16_t __gnu_parallel::_ThreadIndex`

Unsigned integer to index a thread number. The maximum thread number (for each processor) must fit into this type.

Definition at line 123 of file `types.h`.

4.6.3 Enumeration Type Documentation

4.6.3.1 `enum __gnu_parallel::_AlgorithmStrategy`

Strategies for run-time algorithm selection:

Definition at line 67 of file `types.h`.

4.6.3.2 enum `__gnu_parallel::_FindAlgorithm`

Find algorithms:

Definition at line 106 of file types.h.

4.6.3.3 enum `__gnu_parallel::_MultiwayMergeAlgorithm`

Merging algorithms:

Definition at line 85 of file types.h.

4.6.3.4 enum `__gnu_parallel::_Parallelism`

Run-time equivalents for the compile-time tags.

Enumerator

sequential Not parallel.

parallel_unbalanced Parallel unbalanced (equal-sized chunks).

parallel_balanced Parallel balanced (work-stealing).

parallel_omp_loop Parallel with OpenMP dynamic load-balancing.

parallel_omp_loop_static Parallel with OpenMP static load-balancing.

parallel_taskqueue Parallel with OpenMP taskqueue construct.

Definition at line 44 of file types.h.

4.6.3.5 enum `__gnu_parallel::_PartialSumAlgorithm`

Partial sum algorithms: recursive, linear.

Definition at line 91 of file types.h.

4.6.3.6 enum `__gnu_parallel::_SortAlgorithm`

Sorting algorithms:

Definition at line 76 of file types.h.

4.6.3.7 enum `__gnu_parallel::_SplittingAlgorithm`

Sorting/merging algorithms: sampling, `__exact`.

Definition at line 98 of file types.h.

4.6.4 Function Documentation

4.6.4.1 `template<typename _RAIter, typename _DifferenceTp> void __gnu_parallel::_calc_borders (_RAIter __elements, _DifferenceTp __length, _DifferenceTp * __off)`

Precalculate `__advances` for Knuth-Morris-Pratt algorithm.

Parameters

<code>__elements</code>	Begin iterator of sequence to search for.
<code>__length</code>	Length of sequence to search for.
<code>__off</code>	Returned <code>__offsets</code> .

Definition at line 51 of file `search.h`.

Referenced by `__search_template()`.

4.6.4.2 `template<typename _Tp> bool __gnu_parallel::__compare_and_swap (volatile _Tp * __ptr, _Tp __comparand, _Tp __replacement) [inline]`

Compare-and-swap.

Compare `*__ptr` and `__comparand`. If equal, let `*__ptr=__replacement` and return true, return false otherwise.

Parameters

<code>__ptr</code>	Pointer to signed integer.
<code>__comparand</code>	Compare value.
<code>__replacement</code>	Replacement value.

Definition at line 108 of file `parallel/compatibility.h`.

Referenced by `__parallel_partition()`, `__gnu_parallel::RestrictedBoundedConcurrentQueue< pair< _RAIter, _RAIter > >::pop_back()`, and `__gnu_parallel::RestrictedBoundedConcurrentQueue< pair< _RAIter, _RAIter > >::pop_↵front()`.

4.6.4.3 `void __gnu_parallel::__decode2 (_CASable __x, int & __a, int & __b) [inline]`

Decode two integers from one `gnu_parallel::CASable`.

Parameters

<code>↵ __x</code>	<code>__gnu_parallel::CASable</code> to decode integers from.
<code>↵ __a</code>	First integer, to be decoded from the most-significant <code>_CASable_bits/2</code> bits of <code>__x</code> .
<code>↵ __b</code>	Second integer, to be encoded in the least-significant <code>_CASable_bits/2</code> bits of <code>__x</code> .

See also

`__encode2`

Definition at line 133 of file `parallel/base.h`.

References `_CASable_bits`, and `_CASable_mask`.

Referenced by `__gnu_parallel::_RestrictedBoundedConcurrentQueue< pair< _RAIter, _RAIter > >::pop_back()`, `__gnu_parallel::_RestrictedBoundedConcurrentQueue< pair< _RAIter, _RAIter > >::pop_front()`, and `__gnu_parallel::_RestrictedBoundedConcurrentQueue< pair< _RAIter, _RAIter > >::push_front()`.

4.6.4.4 `template<typename _RAIter, typename _DifferenceTp> void __gnu_parallel::_determine_samples (_PMWMSSortingData< _RAIter > * __sd, _DifferenceTp __num_samples)`

Select `_M_samples` from a sequence.

Parameters

<code>__sd</code>	Pointer to algorithm data. Result will be placed in <code>__sd->_M_samples</code> .
<code>__num_samples</code>	Number of <code>_M_samples</code> to select.

Definition at line 97 of file `multiway_mergesort.h`.

References `__equally_split()`, `__gnu_parallel::_PMWMSSortingData< _RAIter >::_M_samples`, `__gnu_parallel::_PMWMSSortingData< _RAIter >::_M_source`, and `__gnu_parallel::_PMWMSSortingData< _RAIter >::_M_starts`.

4.6.4.5 `_CASable __gnu_parallel::_encode2 (int __a, int __b) [inline]`

Encode two integers into one `gnu_parallel::_CASable`.

Parameters

<code>__a</code>	First integer, to be encoded in the most-significant <code>_CASable_bits/2</code> bits.
<code>__b</code>	Second integer, to be encoded in the least-significant <code>_CASable_bits/2</code> bits.

Returns

value encoding `__a` and `__b`.

See also

`__decode2`

Definition at line 119 of file `parallel/base.h`.

References `_CASable_bits`.

Referenced by `__gnu_parallel::_RestrictedBoundedConcurrentQueue< pair< _RAIter, _RAIter > >::RestrictedBoundedConcurrentQueue()`, `__gnu_parallel::_RestrictedBoundedConcurrentQueue< pair< _RAIter, _RAIter > >::pop_back()`, `__gnu_parallel::_RestrictedBoundedConcurrentQueue< pair< _RAIter, _RAIter > >::pop_front()`, and `__gnu_parallel::_RestrictedBoundedConcurrentQueue< pair< _RAIter, _RAIter > >::push_front()`.

4.6.4.6 `template<typename _DifferenceType, typename _OutputIterator> _OutputIterator __gnu_parallel::__equally_split (_DifferenceType __n, _ThreadIndex __num_threads, _OutputIterator __s)`

function to split a sequence into parts of almost equal size.

The resulting sequence `__s` of length `__num_threads+1` contains the splitting positions when splitting the range `[0, __n)` into parts of almost equal size (plus minus 1). The first entry is 0, the last one `n`. There may result empty parts.

Parameters

<code>__n</code>	Number of elements
<code>__num_threads</code>	Number of parts
<code>__s</code>	Splitters

Returns

End of `__splitter` sequence, i.e. `__s+__num_threads+1`

Definition at line 48 of file `equally_split.h`.

Referenced by `__determine_samples()`, `__find_template()`, `__parallel_partial_sum_linear()`, `__parallel_unique_copy()`, `__search_template()`, and `multiway_merge_exact_splitting()`.

4.6.4.7 `template<typename _DifferenceType> _DifferenceType __gnu_parallel::__equally_split_point (_DifferenceType __n, _ThreadIndex __num_threads, _ThreadIndex __thread_no)`

function to split a sequence into parts of almost equal size.

Returns the position of the splitting point between thread number `__thread_no` (included) and thread number `__thread_no+1` (excluded).

Parameters

<code>__n</code>	Number of elements
<code>__num_threads</code>	Number of parts
<code>__thread_no</code>	Number of threads

Returns

splitting point

Definition at line 75 of file `equally_split.h`.

Referenced by `__for_each_template_random_access_ed()`.

4.6.4.8 `template<typename _Tp> _Tp __gnu_parallel::__fetch_and_add (volatile _Tp * __ptr, _Tp __addend) [inline]`

Add a value to a variable, atomically.

Parameters

<code>__ptr</code>	Pointer to a signed integer.
<code>__addend</code>	Value to add.

Definition at line 74 of file `parallel/compatibility.h`.

Referenced by `__parallel_partition()`, and `__gnu_parallel::_RestrictedBoundedConcurrentQueue< pair< _RAIter, _RAIter > >::push_front()`.

```
4.6.4.9 template<typename _RAIter1, typename _RAIter2, typename _Pred, typename _Selector > std::pair<_RAIter1,
    _RAIter2> __gnu_parallel::_find_template ( _RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred,
    _Selector __selector ) [inline]
```

Parallel `std::find`, switch for different algorithms.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence. Must have same length as first sequence.
<code>__pred</code>	Find predicate.
<code>__selector</code>	_Functionality (e. g. <code>std::find_if()</code> , <code>std::equal()</code> ,...)

Returns

Place of finding in both sequences.

Definition at line 60 of file `find.h`.

References `__gnu_parallel::_Settings::get()`, and `std::make_pair()`.

```
4.6.4.10 template<typename _RAIter1, typename _RAIter2, typename _Pred, typename _Selector > std::pair<_RAIter1,
    _RAIter2> __gnu_parallel::_find_template ( _RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred,
    _Selector __selector, equal_split_tag )
```

Parallel `std::find`, equal splitting variant.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence. Second __sequence must have same length as first sequence.
<code>__pred</code>	Find predicate.
<code>__selector</code>	_Functionality (e. g. <code>std::find_if()</code> , <code>std::equal()</code> ,...)

Returns

Place of finding in both sequences.

Definition at line 97 of file `find.h`.

References `__equally_split()`, and `_GLIBCXX_CALL`.

```
4.6.4.11 template<typename _RAIter1, typename _RAIter2, typename _Pred, typename _Selector> std::pair<_RAIter1,
    _RAIter2> __gnu_parallel::__find_template ( _RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred,
    _Selector __selector, growing_blocks_tag )
```

Parallel `std::find`, growing block size variant.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence. Second <code>__sequence</code> must have same length as first sequence.
<code>__pred</code>	Find predicate.
<code>__selector</code>	<code>_Functionality</code> (e. g. <code>std::find_if()</code> , <code>std::equal()</code> ,...)

Returns

Place of finding in both sequences.

See also

`__gnu_parallel::_Settings::find_sequential_search_size`
`__gnu_parallel::_Settings::find_scale_factor`

There are two main differences between the growing blocks and the constant-size blocks variants. 1. For GB, the block size grows; for CSB, the block size is fixed. 2. For GB, the blocks are allocated dynamically; for CSB, the blocks are allocated in a predetermined manner, namely spacial round-robin.

Definition at line 185 of file `find.h`.

References `_GLIBCXX_CALL`, `__gnu_parallel::_Settings::find_scale_factor`, `__gnu_parallel::_Settings::find_sequential_search_size`, and `__gnu_parallel::_Settings::get()`.

```
4.6.4.12 template<typename _RAIter1, typename _RAIter2, typename _Pred, typename _Selector> std::pair<_RAIter1,
    _RAIter2> __gnu_parallel::__find_template ( _RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred,
    _Selector __selector, constant_size_blocks_tag )
```

Parallel `std::find`, constant block size variant.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence. Second <code>__sequence</code> must have same length as first sequence.
<code>__pred</code>	Find predicate.
<code>__selector</code>	<code>_Functionality</code> (e. g. <code>std::find_if()</code> , <code>std::equal()</code> ,...)

Returns

Place of finding in both sequences.

See also

`__gnu_parallel::Settings::find_sequential_search_size`
`__gnu_parallel::Settings::find_block_size` There are two main differences between the growing blocks and the constant-size blocks variants. 1. For GB, the block size grows; for CSB, the block size is fixed. 2. For GB, the blocks are allocated dynamically; for CSB, the blocks are allocated in a predetermined manner, namely spacial round-robin.

Definition at line 315 of file find.h.

References `_GLIBCXX_CALL`, `__gnu_parallel::Settings::find_initial_block_size`, `__gnu_parallel::Settings::find_sequential_search_size`, and `__gnu_parallel::Settings::get()`.

4.6.4.13 `template<typename _Iter, typename _UserOp, typename _Functionality, typename _Red, typename _Result > _UserOp
__gnu_parallel::for_each_template_random_access (_Iter __begin, _Iter __end, _UserOp __user_op, _Functionality &
__functionality, _Red __reduction, _Result __reduction_start, _Result & __output, typename std::iterator_traits< _Iter
>::difference_type __bound, _Parallelism __parallelism_tag)`

Chose the desired algorithm by evaluating `__parallelism_tag`.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__user_op</code>	A user-specified functor (comparator, predicate, associative operator,...)
<code>__functionality</code>	functor to <i>process</i> an element with <code>__user_op</code> (depends on desired functionality, e. g. accumulate, <code>for_each</code> ,...)
<code>__reduction</code>	Reduction functor.
<code>__reduction_start</code>	Initial value for reduction.
<code>__output</code>	Output iterator.
<code>__bound</code>	Maximum number of elements processed.
<code>__parallelism_tag</code>	Parallelization method

Definition at line 61 of file `for_each.h`.

References `__for_each_template_random_access_ed()`, `__for_each_template_random_access_omp_loop()`, `__for_each_template_random_access_workstealing()`, `parallel_omp_loop`, `parallel_omp_loop_static`, and `parallel_unbalanced`.

4.6.4.14 `template<typename _RAIter, typename _Op, typename _Fu, typename _Red, typename _Result > _Op
__gnu_parallel::for_each_template_random_access_ed (_RAIter __begin, _RAIter __end, _Op __o, _Fu & __f, _Red __r,
_Result __base, _Result & __output, typename std::iterator_traits< _RAIter >::difference_type __bound)`

Embarrassingly parallel algorithm for random access iterators, using hand-crafted parallelization by equal splitting the work.

Parameters

<code>__begin</code>	Begin iterator of element sequence.
<code>__end</code>	End iterator of element sequence.
<code>__o</code>	User-supplied functor (comparator, predicate, adding functor, ...)
<code>__f</code>	Functor to "process" an element with <code>__op</code> (depends on desired functionality, e. g. for <code>std::for_each()</code> , ...).
<code>__r</code>	Functor to "add" a single <code>__result</code> to the already processed elements (depends on functionality).
<code>__base</code>	Base value for reduction.
<code>__output</code>	Pointer to position where final result is written to
<code>__bound</code>	Maximum number of elements processed (e. g. for <code>std::count_n()</code>).

Returns

User-supplied functor (that may contain a part of the result).

Definition at line 67 of file `par_loop.h`.

References `__equally_split_point()`.

Referenced by `__for_each_template_random_access()`.

4.6.4.15 `template<typename _RAIter, typename _Op, typename _Fu, typename _Red, typename _Result > _Op
__gnu_parallel::__for_each_template_random_access_omp_loop(_RAIter __begin, _RAIter __end, _Op __o, _Fu & __f,
_Red __r, _Result __base, _Result & __output, typename std::iterator_traits< _RAIter >::difference_type __bound)`

Embarrassingly parallel algorithm for random access iterators, using an OpenMP for loop.

Parameters

<code>__begin</code>	Begin iterator of element sequence.
<code>__end</code>	End iterator of element sequence.
<code>__o</code>	User-supplied functor (comparator, predicate, adding functor, etc.).
<code>__f</code>	Functor to <i>process</i> an element with <code>__op</code> (depends on desired functionality, e. g. for <code>std::for_each()</code> , ...).
<code>__r</code>	Functor to <i>add</i> a single <code>__result</code> to the already processed elements (depends on functionality).
<code>__base</code>	Base value for reduction.
<code>__output</code>	Pointer to position where final result is written to
<code>__bound</code>	Maximum number of elements processed (e. g. for <code>std::count_n()</code>).

Returns

User-supplied functor (that may contain a part of the result).

Definition at line 67 of file `omp_loop.h`.

Referenced by `__for_each_template_random_access()`.

4.6.4.16 `template<typename _RAIter , typename _Op , typename _Fu , typename _Red , typename _Result > _Op
__gnu_parallel::__for_each_template_random_access_omp_loop_static (_RAIter __begin, _RAIter __end, _Op __o, _Fu &
__f, _Red __r, _Result __base, _Result & __output, typename std::iterator_traits< _RAIter >::difference_type __bound)`

Embarrassingly parallel algorithm for random access iterators, using an OpenMP for loop with static scheduling.

Parameters

<code>__begin</code>	Begin iterator of element sequence.
<code>__end</code>	End iterator of element sequence.
<code>__o</code>	User-supplied functor (comparator, predicate, adding functor, ...).
<code>__f</code>	Functor to <i>process</i> an element with <code>__op</code> (depends on desired functionality, e. g. for <code>std::for_each()</code> , ...).
<code>__r</code>	Functor to <i>add</i> a single <code>__result</code> to the already processed <code>__elements</code> (depends on functionality).
<code>__base</code>	Base value for reduction.
<code>__output</code>	Pointer to position where final result is written to
<code>__bound</code>	Maximum number of elements processed (e. g. for <code>std::count_n()</code>).

Returns

User-supplied functor (that may contain a part of the result).

Definition at line 66 of file `omp_loop_static.h`.

4.6.4.17 `template<typename _RAIter , typename _Op , typename _Fu , typename _Red , typename _Result > _Op
__gnu_parallel::__for_each_template_random_access_workstealing (_RAIter __begin, _RAIter __end, _Op __op, _Fu &
__f, _Red __r, _Result __base, _Result & __output, typename std::iterator_traits< _RAIter >::difference_type __bound)`

Work stealing algorithm for random access iterators.

Uses $O(1)$ additional memory. Synchronization at job lists is done with atomic operations.

Parameters

<code>__begin</code>	Begin iterator of element sequence.
<code>__end</code>	End iterator of element sequence.
<code>__op</code>	User-supplied functor (comparator, predicate, adding functor, ...).
<code>__f</code>	Functor to <i>process</i> an element with <code>__op</code> (depends on desired functionality, e. g. for <code>std::for_each()</code> , ...).
<code>__r</code>	Functor to <i>add</i> a single <code>__result</code> to the already processed elements (depends on functionality).
<code>__base</code>	Base value for reduction.
<code>__output</code>	Pointer to position where final result is written to
<code>__bound</code>	Maximum number of elements processed (e. g. for <code>std::count_n()</code>).

Returns

User-supplied functor (that may contain a part of the result).

Definition at line 99 of file `workstealing.h`.

References `__yield()`, `_GLIBCXX_CALL`, `__gnu_parallel::_Job<_DifferenceTp>::_M_first`, `__gnu_parallel::_Job<_DifferenceTp>::_M_last`, `__gnu_parallel::_Job<_DifferenceTp>::_M_load`, `__gnu_parallel::_Settings::cache_line_size`, `__gnu_parallel::_Settings::get()`, and `min()`.

Referenced by `__for_each_template_random_access()`.

4.6.4.18 `template<typename _Iter, typename _Compare> bool __gnu_parallel::__is_sorted (_Iter __begin, _Iter __end, _Compare __comp)`

Check whether `[__begin, __end)` is sorted according to `__comp`.

Parameters

<code>__begin</code>	Begin iterator of sequence.
<code>__end</code>	End iterator of sequence.
<code>__comp</code>	Comparator.

Returns

`true` if sorted, `false` otherwise.

Definition at line 51 of file `checkers.h`.

Referenced by `__sequential_multiway_merge()`, `multiway_merge_loser_tree_sentinel()`, and `parallel_multiway_merge()`.

4.6.4.19 `template<typename _RAIter, typename _Compare> _RAIter __gnu_parallel::__median_of_three_iterators (_RAIter __a, _RAIter __b, _RAIter __c, _Compare __comp)`

Compute the median of three referenced elements, according to `__comp`.

Parameters

<code>__a</code>	First iterator.
<code>__b</code>	Second iterator.
<code>__c</code>	Third iterator.
<code>__comp</code>	Comparator.

Definition at line 398 of file `parallel/base.h`.

Referenced by `__qsb_divide()`.

4.6.4.20 `template<typename _RAIter1, typename _RAIter2, typename _OutputIterator, typename _DifferenceTp, typename _Compare> _OutputIterator __gnu_parallel::__merge_advance (_RAIter1 &__begin1, _RAIter1 __end1, _RAIter2 &__begin2, _RAIter2 __end2, _OutputIterator __target, _DifferenceTp __max_length, _Compare __comp) [inline]`

Merge routine being able to merge only the `__max_length` smallest elements.

The `__begin` iterators are advanced accordingly, they might not reach `__end`, in contrast to the usual variant. Static switch on whether to use the conditional-move variant.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence.
<code>__end2</code>	End iterator of second sequence.
<code>__target</code>	Target begin iterator.
<code>__max_length</code>	Maximum number of elements to merge.
<code>__comp</code>	Comparator.

Returns

Output end iterator.

Definition at line 171 of file `merge.h`.

References `__merge_advance_movc()`, and `_GLIBCXX_CALL`.

Referenced by `__parallel_merge_advance()`, and `__sequential_multiway_merge()`.

4.6.4.21 `template<typename _RAIter1, typename _RAIter2, typename _OutputIterator, typename _DifferenceTp, typename _Compare> _OutputIterator __gnu_parallel::__merge_advance_movc (_RAIter1 & __begin1, _RAIter1 __end1, _RAIter2 & __begin2, _RAIter2 __end2, _OutputIterator __target, _DifferenceTp __max_length, _Compare __comp)`

Merge routine being able to merge only the `__max_length` smallest elements.

The `__begin` iterators are advanced accordingly, they might not reach `__end`, in contrast to the usual variant. Specially designed code should allow the compiler to generate conditional moves instead of branches.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence.
<code>__end2</code>	End iterator of second sequence.
<code>__target</code>	Target begin iterator.
<code>__max_length</code>	Maximum number of elements to merge.
<code>__comp</code>	Comparator.

Returns

Output end iterator.

Definition at line 105 of file `merge.h`.

Referenced by `__merge_advance()`.

4.6.4.22 `template<typename _RAIter1 , typename _RAIter2 , typename _OutputIterator , typename _DifferenceTp , typename _Compare > _OutputIterator __gnu_parallel::__merge_advance_usual (_RAIter1 & __begin1, _RAIter1 __end1, _RAIter2 & __begin2, _RAIter2 __end2, _OutputIterator __target, _DifferenceTp __max_length, _Compare __comp)`

Merge routine being able to merge only the `__max_length` smallest elements.

The `__begin` iterators are advanced accordingly, they might not reach `__end`, in contrast to the usual variant.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence.
<code>__end2</code>	End iterator of second sequence.
<code>__target</code>	Target begin iterator.
<code>__max_length</code>	Maximum number of elements to merge.
<code>__comp</code>	Comparator.

Returns

Output end iterator.

Definition at line 57 of file `merge.h`.

4.6.4.23 `template<typename _RAIter1 , typename _RAIter2 , typename _RAIter3 , typename _Compare > _RAIter3 __gnu_parallel::__parallel_merge_advance (_RAIter1 & __begin1, _RAIter1 __end1, _RAIter2 & __begin2, _RAIter2 __end2, _RAIter3 __target, typename std::iterator_traits< _RAIter1 >::difference_type __max_length, _Compare __comp) [inline]`

Merge routine fallback to sequential in case the iterators of the two input sequences are of different type.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence.
<code>__end2</code>	End iterator of second sequence.
<code>__target</code>	Target begin iterator.
<code>__max_length</code>	Maximum number of elements to merge.
<code>__comp</code>	Comparator.

Returns

Output end iterator.

Definition at line 195 of file `merge.h`.

References `__merge_advance()`.

4.6.4.24 `template<typename _RAIter1, typename _RAIter3, typename _Compare> _RAIter3 __gnu_parallel::__parallel_merge(↵
advance (_RAIter1 & __begin1, _RAIter1 __end1, _RAIter1 & __begin2, _RAIter1 __end2, _RAIter3 __target, typename
std::iterator_traits< _RAIter1 >::difference_type __max_length, _Compare __comp) [inline]`

Parallel merge routine being able to merge only the `__max_length` smallest elements.

The `__begin` iterators are advanced accordingly, they might not reach `__end`, in contrast to the usual variant. The functionality is projected onto `parallel_multiway_merge`.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence.
<code>__end2</code>	End iterator of second sequence.
<code>__target</code>	Target begin iterator.
<code>__max_length</code>	Maximum number of elements to merge.
<code>__comp</code>	Comparator.

Returns

Output end iterator.

Definition at line 223 of file `merge.h`.

References `std::make_pair()`, `multiway_merge_exact_splitting()`, and `parallel_multiway_merge()`.

4.6.4.25 `template<typename _RAIter, typename _Compare> void __gnu_parallel::__parallel_nth_element (_RAIter __begin,
_RAIter __nth, _RAIter __end, _Compare __comp)`

Parallel implementation of `std::nth_element()`.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__nth</code>	_Iterator of element that must be in position afterwards.
<code>__end</code>	End iterator of input sequence.
<code>__comp</code>	Comparator.

Definition at line 332 of file `partition.h`.

References `__parallel_partition()`, `_GLIBCXX_CALL`, `__gnu_parallel::_Settings::get()`, `std::max()`, `__gnu_parallel::_↵
Settings::nth_element_minimal_n`, and `__gnu_parallel::_Settings::partition_minimal_n`.

Referenced by `__parallel_partial_sort()`.

4.6.4.26 `template<typename _RAIter, typename _Compare> void __gnu_parallel::__parallel_partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end, _Compare __comp)`

Parallel implementation of `std::partial_sort()`.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__middle</code>	Sort until this position.
<code>__end</code>	End iterator of input sequence.
<code>__comp</code>	Comparator.

Definition at line 422 of file `partition.h`.

References `__parallel_nth_element()`.

```
4.6.4.27 template<typename _Iter , typename _OutputIterator , typename _BinaryOperation > _OutputIterator
    __gnu_parallel::__parallel_partial_sum ( _Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __bin_op
    )
```

Parallel partial sum front-`__end`.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__result</code>	Begin iterator of output sequence.
<code>__bin_op</code>	Associative binary function.

Returns

End iterator of output sequence.

Definition at line 205 of file `partial_sum.h`.

References `__parallel_partial_sum_linear()`, `_GLIBCXX_CALL`, and `__gnu_parallel::_Settings::get()`.

```
4.6.4.28 template<typename _Iter , typename _OutputIterator , typename _BinaryOperation > _OutputIterator
    __gnu_parallel::__parallel_partial_sum_basecase ( _Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation
    __bin_op, typename std::iterator_traits< _Iter >::value_type __value )
```

Base case prefix sum routine.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__result</code>	Begin iterator of output sequence.
<code>__bin_op</code>	Associative binary function.
<code>__value</code>	Start value. Must be passed since the neutral element is unknown in general.

Returns

End iterator of output sequence.

Definition at line 58 of file `partial_sum.h`.

Referenced by `__parallel_partial_sum_linear()`.

```
4.6.4.29 template<typename _Iter , typename _OutputIterator , typename _BinaryOperation > _OutputIterator
    __gnu_parallel::__parallel_partial_sum_linear ( _Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation
    __bin_op, typename std::iterator_traits< _Iter >::difference_type __n )
```

Parallel partial sum implementation, two-phase approach, no recursion.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__result</code>	Begin iterator of output sequence.
<code>__bin_op</code>	Associative binary function.
<code>__n</code>	Length of sequence.

Returns

End iterator of output sequence.

Definition at line 89 of file `partial_sum.h`.

References `__equally_split()`, `__parallel_partial_sum_basecase()`, `std::accumulate()`, `__gnu_parallel::_Settings::get()`, and `__gnu_parallel::_Settings::partial_sum_dilation`.

Referenced by `__parallel_partial_sum()`.

```
4.6.4.30 template<typename _RAlter , typename _Predicate > std::iterator_traits<_RAlter>::difference_type
    __gnu_parallel::__parallel_partition ( _RAlter __begin, _RAlter __end, _Predicate __pred, _ThreadIndex __num_threads
    )
```

Parallel implementation of `std::partition`.

Parameters

<code>__begin</code>	Begin iterator of input sequence to split.
<code>__end</code>	End iterator of input sequence to split.
<code>__pred</code>	Partition predicate, possibly including some kind of pivot.
<code>__num_threads</code>	Maximum number of threads to use for this task.

Returns

Number of elements not fulfilling the predicate.

Definition at line 56 of file partition.h.

References `__compare_and_swap()`, `__fetch_and_add()`, `_GLIBCXX_CALL`, `_GLIBCXX_VOLATILE`, `__gnu_parallel::Settings::get()`, `__gnu_parallel::Settings::partition_chunk_share`, and `__gnu_parallel::Settings::partition_chunk_size`.

Referenced by `__parallel_nth_element()`, `__parallel_sort_qs_divide()`, and `__qsb_divide()`.

4.6.4.31 `template<typename _RAIter, typename _RandomNumberGenerator> void __gnu_parallel::parallel_random_shuffle (_RAIter __begin, _RAIter __end, _RandomNumberGenerator __rng = _RandomNumber()) [inline]`

Parallel random public call.

Parameters

<code>__begin</code>	Begin iterator of sequence.
<code>__end</code>	End iterator of sequence.
<code>__rng</code>	Random number generator to use.

Definition at line 522 of file random_shuffle.h.

References `__parallel_random_shuffle_drs()`.

4.6.4.32 `template<typename _RAIter, typename _RandomNumberGenerator> void __gnu_parallel::parallel_random_shuffle_drs (_RAIter __begin, _RAIter __end, typename std::iterator_traits<_RAIter>::difference_type __n, _ThreadIndex __num_threads, _RandomNumberGenerator & __rng)`

Main parallel random shuffle step.

Parameters

<code>__begin</code>	Begin iterator of sequence.
<code>__end</code>	End iterator of sequence.
<code>__n</code>	Length of sequence.
<code>__num_threads</code>	Number of threads to use.
<code>__rng</code>	Random number generator to use.

Definition at line 265 of file random_shuffle.h.

References `__gnu_parallel::DRSSorterPU<_RAIter, _RandomNumberGenerator>::__bins_end`, `__parallel_random_shuffle_drs_pu()`, `__rd_log2()`, `__round_up_to_pow2()`, `__sequential_random_shuffle()`, `_GLIBCXX_CALL`, `__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::__M_bin_proc`, `__gnu_parallel::DRSSorterPU<_RAIter, _RandomNumberGenerator>::__M_bins_begin`, `__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::__M_dist`,

`__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::M_num_bins`, `__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::M_num_bits`, `__gnu_parallel::DRSSorterPU<_RAIter, _RandomNumberGenerator>::M_num_threads`, `__gnu_parallel::DRSSorterPU<_RAIter, _RandomNumberGenerator>::M_sd`, `__gnu_parallel::DRSSorterPU<_RAIter, _RandomNumberGenerator>::M_seed`, `__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::M_starts`, `__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::M_temporaries`, `__gnu_parallel::Settings::get()`, `__gnu_parallel::Settings::L2_cache_size`, `std::min()`, and `__gnu_parallel::Settings::TLB_size`.

Referenced by `__parallel_random_shuffle()`.

4.6.4.33 `template<typename _RAIter, typename _RandomNumberGenerator> void __gnu_parallel::parallel_random_shuffle_drs_pu (_DRSSorterPU<_RAIter, _RandomNumberGenerator> * __pus)`

Random shuffle code executed by each thread.

Parameters

<code>__pus</code>	Array of thread-local data records.
--------------------	-------------------------------------

Definition at line 122 of file `random_shuffle.h`.

References `__random_number_pow2()`, `__sequential_random_shuffle()`, `__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::M_dist`, `__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::M_num_bins`, `__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::M_num_bits`, `__gnu_parallel::DRSSorterPU<_RAIter, _RandomNumberGenerator>::M_num_threads`, `__gnu_parallel::DRSSorterPU<_RAIter, _RandomNumberGenerator>::M_sd`, `__gnu_parallel::DRSSorterPU<_RAIter, _RandomNumberGenerator>::M_seed`, `__gnu_parallel::DRandomShufflingGlobalData<_RAIter>::M_starts`, and `std::partial_sum()`.

Referenced by `__parallel_random_shuffle_drs()`.

4.6.4.34 `template<bool __stable, typename _RAIter, typename _Compare> void __gnu_parallel::parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, multiway_mergesort_tag __parallelism) [inline]`

Choose multiway mergesort, splitting variant at run-time, for parallel sorting.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__comp</code>	Comparator.

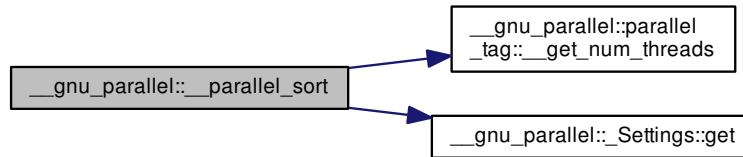
Template Parameters

<code>__stable</code>	Sort stable.
-----------------------	--------------

Definition at line 75 of file `sort.h`.

References `__gnu_parallel::parallel_tag::get_num_threads()`, `_GLIBCXX_CALL`, and `__gnu_parallel::Settings::get()`.

Here is the call graph for this function:



4.6.4.35 `template<bool __stable, typename _RAIter, typename _Compare> void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, multiway_mergesort_exact_tag __parallelism) [inline]`

Choose multiway mergesort with exact splitting, for parallel sorting.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__comp</code>	Comparator.

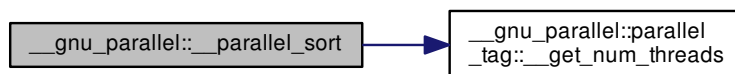
Template Parameters

<code>__stable</code>	Sort stable.
-----------------------	--------------

Definition at line 99 of file sort.h.

References `__gnu_parallel::parallel_tag::__get_num_threads()`, and `_GLIBCXX_CALL`.

Here is the call graph for this function:



4.6.4.36 `template<bool __stable, typename _RAIter, typename _Compare> void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, multiway_mergesort_sampling_tag __parallelism) [inline]`

Choose multiway mergesort with splitting by sampling, for parallel sorting.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__comp</code>	Comparator.

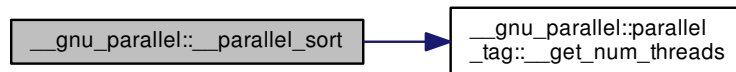
Template Parameters

<code>__stable</code>	Sort stable.
-----------------------	--------------

Definition at line 120 of file `sort.h`.

References `__gnu_parallel::parallel_tag::__get_num_threads()`, and `_GLIBCXX_CALL`.

Here is the call graph for this function:



4.6.4.37 `template<bool __stable, typename _RAIter, typename _Compare> void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, quicksort_tag __parallelism) [inline]`

Choose quicksort for parallel sorting.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__comp</code>	Comparator.

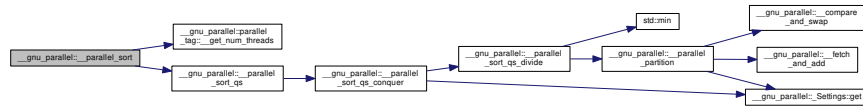
Template Parameters

<code>__stable</code>	Sort stable.
-----------------------	--------------

Definition at line 140 of file `sort.h`.

References `__gnu_parallel::parallel_tag::__get_num_threads()`, `__parallel_sort_qs()`, and `_GLIBCXX_CALL`.

Here is the call graph for this function:



4.6.4.38 `template<bool __stable, typename _RAIter, typename _Compare> void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, balanced_quicksort_tag __parallelism) [inline]`

Choose balanced quicksort for parallel sorting.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__comp</code>	Comparator.

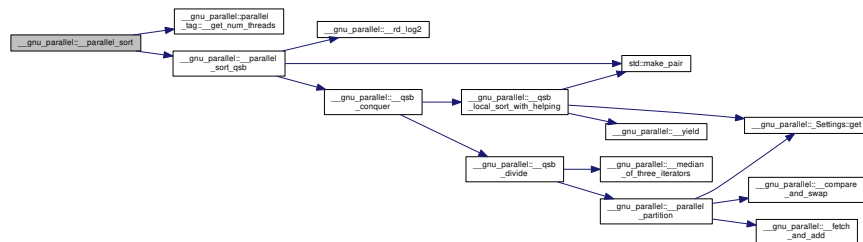
Template Parameters

<code>__stable</code>	Sort stable.
-----------------------	--------------

Definition at line 161 of file sort.h.

References `__gnu_parallel::parallel_tag::__get_num_threads()`, `__parallel_sort_qsb()`, and `_GLIBCXX_CALL`.

Here is the call graph for this function:



4.6.4.39 `template<bool __stable, typename _RAIter, typename _Compare> void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, default_parallel_tag __parallelism) [inline]`

Choose multiway mergesort with exact splitting, for parallel sorting.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__comp</code>	Comparator.

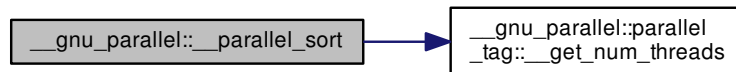
Template Parameters

<code>__stable</code>	Sort stable.
-----------------------	--------------

Definition at line 183 of file `sort.h`.

References `__gnu_parallel::parallel_tag::__get_num_threads()`, and `_GLIBCXX_CALL`.

Here is the call graph for this function:



4.6.4.40 `template<bool __stable, typename _RAIter, typename _Compare> void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, parallel_tag __parallelism) [inline]`

Choose a parallel sorting algorithm.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__comp</code>	Comparator.

Template Parameters

<code>__stable</code>	Sort stable.
-----------------------	--------------

Definition at line 203 of file `sort.h`.

References `__gnu_parallel::parallel_tag::__get_num_threads()`, `__parallel_sort_qs()`, `__parallel_sort_qsb()`, `_GLIBCXX_CALL`, and `__gnu_parallel::_Settings::get()`.

References `__parallel_sort_qs_divide()`, and `__gnu_parallel::_Settings::get()`.

Referenced by `__parallel_sort_qs()`.

4.6.4.43 `template<typename _RAIter, typename _Compare> std::iterator_traits<_RAIter>::difference_type
__gnu_parallel::__parallel_sort_qs_divide (_RAIter __begin, _RAIter __end, _Compare __comp, typename
std::iterator_traits<_RAIter>::difference_type __pivot_rank, typename std::iterator_traits<_RAIter>::difference_type
__num_samples, _ThreadIndex __num_threads)`

Unbalanced quicksort divide step.

Parameters

<code>__begin</code>	Begin iterator of subsequence.
<code>__end</code>	End iterator of subsequence.
<code>__comp</code>	Comparator.
<code>__pivot_rank</code>	Desired <code>__rank</code> of the pivot.
<code>__num_samples</code>	Choose pivot from that many samples.
<code>__num_threads</code>	Number of threads that are allowed to work on this part.

Definition at line 51 of file `quicksort.h`.

References `__parallel_partition()`, and `std::min()`.

Referenced by `__parallel_sort_qs_conquer()`.

4.6.4.44 `template<typename _RAIter, typename _Compare> void __gnu_parallel::__parallel_sort_qsb (_RAIter __begin, _RAIter
__end, _Compare __comp, _ThreadIndex __num_threads)`

Top-level quicksort routine.

Parameters

<code>__begin</code>	Begin iterator of sequence.
<code>__end</code>	End iterator of sequence.
<code>__comp</code>	Comparator.
<code>__num_threads</code>	Number of threads that are allowed to work on this part.

Definition at line 433 of file `balanced_quicksort.h`.

References `__qsb_conquer()`, `__rd_log2()`, `_GLIBCXX_CALL`, `__gnu_parallel::QSBThreadLocal<_RAIter>::_M_←
elements_leftover`, `__gnu_parallel::QSBThreadLocal<_RAIter>::_M_leftover_parts`, and `std::make_pair()`.

Referenced by `__parallel_sort()`.

4.6.4.45 `template<typename _Iter , class _OutputIterator , class _BinaryPredicate > _OutputIterator
__gnu_parallel::__parallel_unique_copy (_Iter __first, _Iter __last, _OutputIterator __result, _BinaryPredicate
__binary_pred)`

Parallel `std::unique_copy()`, w/___o explicit equality predicate.

Parameters

<code>__first</code>	Begin iterator of input sequence.
<code>__last</code>	End iterator of input sequence.
<code>__result</code>	Begin iterator of result __sequence.
<code>__binary_pred</code>	Equality predicate.

Returns

End iterator of result __sequence.

Definition at line 50 of file `unique_copy.h`.

References `__equally_split()`, and `_GLIBCXX_CALL`.

Referenced by `__parallel_unique_copy()`.

4.6.4.46 `template<typename _Iter, class _OutputIterator> _OutputIterator __gnu_parallel::__parallel_unique_copy (_Iter __first, _Iter __last, _OutputIterator __result) [inline]`

Parallel `std::unique_copy()`, without explicit equality predicate.

Parameters

<code>__first</code>	Begin iterator of input sequence.
<code>__last</code>	End iterator of input sequence.
<code>__result</code>	Begin iterator of result __sequence.

Returns

End iterator of result __sequence.

Definition at line 186 of file `unique_copy.h`.

References `__parallel_unique_copy()`.

4.6.4.47 `template<typename _RAIter, typename _Compare> void __gnu_parallel::__qsb_conquer (_QSBThreadLocal< _RAIter> ** __tls, _RAIter __begin, _RAIter __end, _Compare __comp, _ThreadIndex __iam, _ThreadIndex __num_threads, bool __parent_wait)`

Quicksort conquer step.

Parameters

<code>__tls</code>	Array of thread-local storages.
<code>__begin</code>	Begin iterator of subsequence.
<code>__end</code>	End iterator of subsequence.
<code>__comp</code>	Comparator.
<code>__num_threads</code>	Number of threads that are allowed to work on this part.

Definition at line 174 of file `balanced_quicksort.h`.

References `__qsb_divide()`, `__qsb_local_sort_with_helping()`, `__gnu_parallel::QSBThreadLocal<_RAIter>::M_elements_leftover`, `__gnu_parallel::QSBThreadLocal<_RAIter>::M_initial`, `std::pair<_T1, _T2>::first`, and `std::pair<_T1, _T2>::second`.

Referenced by `__parallel_sort_qsb()`.

4.6.4.48 `template<typename _RAIter, typename _Compare> std::iterator_traits<_RAIter>::difference_type
__gnu_parallel::__qsb_divide (_RAIter __begin, _RAIter __end, _Compare __comp, _ThreadIndex __num_threads)`

Balanced quicksort divide step.

Parameters

<code>__begin</code>	Begin iterator of subsequence.
<code>__end</code>	End iterator of subsequence.
<code>__comp</code>	Comparator.
<code>__num_threads</code>	Number of threads that are allowed to work on this part.

Precondition

`(__end-__begin)>=1`

Definition at line 103 of file `balanced_quicksort.h`.

References `__median_of_three_iterators()`, and `__parallel_partition()`.

Referenced by `__qsb_conquer()`.

4.6.4.49 `template<typename _RAIter, typename _Compare> void __gnu_parallel::__qsb_local_sort_with_helping (
__QSBThreadLocal<_RAIter> ** __tls, _Compare & __comp, _ThreadIndex __iam, bool __wait)`

Quicksort step doing load-balanced local sort.

Parameters

<code>__tls</code>	Array of thread-local storages.
<code>__comp</code>	Comparator.
<code>__iam</code>	Number of the thread processing this function.

Definition at line 250 of file `balanced_quicksort.h`.

References `__yield()`, `__gnu_parallel::QSBThreadLocal<_RAIter>::M_elements_leftover`, `__gnu_parallel::QSBThreadLocal<_RAIter>::M_initial`, `__gnu_parallel::QSBThreadLocal<_RAIter>::M_leftover_parts`, `__gnu_parallel::QSBThreadLocal<_RAIter>::M_num_threads`, `__gnu_parallel::Settings::get()`, `std::make_pair()`, and `__gnu_parallel::Settings::sort_qsb_base_case_maximal_n`.

Referenced by `__qsb_conquer()`.

4.6.4.50 `template<typename _RandomNumberGenerator > int __gnu_parallel::__random_number_pow2 (int __logp,
_RandomNumberGenerator & __rng) [inline]`

Generate a random number in $[0, 2^{\text{__logp}})$.

Parameters

<code>__logp</code>	Logarithm (basis 2) of the upper range __bound.
<code>__rng</code>	Random number generator to use.

Definition at line 115 of file `random_shuffle.h`.

Referenced by `__parallel_random_shuffle_drs_pu()`, and `__sequential_random_shuffle()`.

4.6.4.51 `template<typename _Size > _Size __gnu_parallel::__rd_log2 (_Size __n) [inline]`

Calculates the rounded-down logarithm of `__n` for base 2.

Parameters

<code>__n</code>	Argument.
------------------	-----------

Returns

Returns 0 for any argument < 1 .

Definition at line 102 of file `parallel/base.h`.

Referenced by `__parallel_random_shuffle_drs()`, `__parallel_sort_qsb()`, `__round_up_to_pow2()`, `__sequential_random_shuffle()`, `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_LoserTreeBase()`, `multiseq_partition()`, and `multiseq_selection()`.

4.6.4.52 `template<typename _Tp > _Tp __gnu_parallel::__round_up_to_pow2 (_Tp __x)`

Round up to the next greater power of 2.

Parameters

<code>__x</code>	_Integer to round up
------------------	----------------------

Definition at line 248 of file `random_shuffle.h`.

References `__rd_log2()`.

Referenced by `__parallel_random_shuffle_drs()`, `__sequential_random_shuffle()`, and `multiseq_selection()`.

4.6.4.53 `template<typename __RAIter1, typename __RAIter2, typename _Pred> __RAIter1 __gnu_parallel::__search_template (__RAIter1 __begin1, __RAIter1 __end1, __RAIter2 __begin2, __RAIter2 __end2, _Pred __pred)`

Parallel `std::search`.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence.
<code>__end2</code>	End iterator of second sequence.
<code>__pred</code>	Find predicate.

Returns

Place of finding in first sequences.

Definition at line 81 of file `search.h`.

References `__calc_borders()`, `__equally_split()`, `_GLIBCXX_CALL`, and `std::min()`.

4.6.4.54 `template<bool __stable, bool __sentinels, typename _RAIterIterator, typename _RAIter3, typename _DifferenceTp, typename _Compare> _RAIter3 __gnu_parallel::__sequential_multiway_merge (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 __target, const typename std::iterator_traits< typename std::iterator_traits< _RAIterIterator >::value_type::first_type >::value_type & __sentinel, _DifferenceTp __length, _Compare __comp)`

Sequential multi-way merging switch.

The `_GLIBCXX_PARALLEL_DECISION` is based on the branching factor and runtime settings.

Parameters

<code>__seqs_begin</code>	Begin iterator of iterator pair input sequence.
<code>__seqs_end</code>	End iterator of iterator pair input sequence.
<code>__target</code>	Begin iterator of output sequence.
<code>__comp</code>	Comparator.
<code>__length</code>	Maximum length to merge, possibly larger than the number of elements available.
<code>__sentinel</code>	The sequences have <code>__a</code> <code>__sentinel</code> element.

Returns

End iterator of output sequence.

Definition at line 920 of file `multiway_merge.h`.

References `__is_sorted()`, `__merge_advance()`, `_GLIBCXX_CALL`, and `_GLIBCXX_PARALLEL_LENGTH`.

Referenced by `multiway_merge()`, and `multiway_merge_sentinels()`.

4.6.4.55 `template<typename _RAIter, typename _RandomNumberGenerator> void __gnu_parallel::__sequential_random_shuffle (_RAIter __begin, _RAIter __end, _RandomNumberGenerator & __rng)`

Sequential cache-efficient random shuffle.

Parameters

<code>__begin</code>	Begin iterator of sequence.
<code>__end</code>	End iterator of sequence.
<code>__rng</code>	Random number generator to use.

Definition at line 410 of file `random_shuffle.h`.

References `__random_number_pow2()`, `__rd_log2()`, `__round_up_to_pow2()`, `__gnu_parallel::_Settings::get()`, `__gnu_parallel::_Settings::L2_cache_size`, `std::min()`, `std::partial_sum()`, and `__gnu_parallel::_Settings::TLB_size`.

Referenced by `__parallel_random_shuffle_drs()`, and `__parallel_random_shuffle_drs_pu()`.

4.6.4.56 `template<typename _Iter> void __gnu_parallel::__shrink (std::vector<_Iter> & __os_starts, size_t & __count_to_two, size_t & __range_length)`

Combines two ranges into one and thus halves the number of ranges.

Parameters

<code>__os_starts</code>	Start positions worked on (oversampled).
<code>__count_to_two</code>	Counts up to 2.
<code>__range_length</code>	Current length of a chunk.

Definition at line 70 of file `list_partition.h`.

References `std::vector<_Tp, _Alloc>::size()`.

Referenced by `__shrink_and_double()`.

4.6.4.57 `template<typename _Iter> void __gnu_parallel::__shrink_and_double (std::vector<_Iter> & __os_starts, size_t & __count_to_two, size_t & __range_length, const bool __make_twice)`

Shrinks and doubles the ranges.

Parameters

<code>__os_starts</code>	Start positions worked on (oversampled).
<code>__count_to_two</code>	Counts up to 2.
<code>__range_length</code>	Current length of a chunk.
<code>__make_twice</code>	Whether the <code>__os_starts</code> is allowed to be grown or not

Definition at line 50 of file list_partition.h.

References `__shrink()`, `std::vector<_Tp, _Alloc >::resize()`, and `std::vector<_Tp, _Alloc >::size()`.

Referenced by `list_partition()`.

4.6.4.58 `void __gnu_parallel::__yield() [inline]`

Yield control to another thread, without waiting for the end of the time slice.

Definition at line 121 of file parallel/compatibility.h.

Referenced by `__for_each_template_random_access_workstealing()`, and `__qsb_local_sort_with_helping()`.

4.6.4.59 `template<typename _Iter, typename _FunctorType> size_t __gnu_parallel::list_partition(const _Iter __begin, const _Iter __end, _Iter * __starts, size_t * __lengths, const int __num_parts, _FunctorType & __f, int __oversampling = 0)`

Splits a sequence given by input iterators into parts of almost equal size.

The function needs only one pass over the sequence.

Parameters

<code>__begin</code>	Begin iterator of input sequence.
<code>__end</code>	End iterator of input sequence.
<code>__starts</code>	Start iterators for the resulting parts, dimension <code>__num_parts+1</code> . For convenience, <code>__starts [__num_parts]</code> contains the end iterator of the sequence.
<code>__lengths</code>	Length of the resulting parts.
<code>__num_parts</code>	Number of parts to split the sequence into.
<code>__f</code>	Functor to be applied to each element by traversing <code>__it</code>
<code>__oversampling</code>	Oversampling factor. If 0, then the partitions will differ in at most $\sqrt{\text{end} - \text{begin}}$ elements. Otherwise, the ratio between the longest and the shortest part is bounded by $1/(\text{oversampling} \cdot \text{num_parts})$

Returns

Length of the whole sequence.

Definition at line 101 of file list_partition.h.

References `__shrink_and_double()`, and `std::vector<_Tp, _Alloc >::size()`.

4.6.4.60 `template<typename _Tp> const _Tp& __gnu_parallel::max(const _Tp & __a, const _Tp & __b) [inline]`

Equivalent to `std::max`.

Definition at line 150 of file parallel/base.h.

4.6.4.61 `template<typename _Tp> const _Tp& __gnu_parallel::min (const _Tp & __a, const _Tp & __b) [inline]`

Equivalent to `std::min`.

Definition at line 144 of file `parallel/base.h`.

Referenced by `__for_each_template_random_access_workstealing()`.

4.6.4.62 `template<typename _RanSeqs , typename _RankType , typename _RankIterator , typename
_Compare > void __gnu_parallel::multiseq_partition (_RanSeqs __begin_seqs, _RanSeqs
__end_seqs, _RankType __rank, _RankIterator __begin_offsets, _Compare __comp =
std::less< typename std::iterator_traits<typename std::iterator<
_traits<_RanSeqs>::value_type:: first_type>::value_type>())`

Splits several sorted sequences at a certain global `__rank`, resulting in a splitting point for each sequence. The sequences are passed via a sequence of random-access iterator pairs, none of the sequences may be empty. If there are several equal elements across the split, the ones on the `__left` side will be chosen from sequences with smaller number.

Parameters

<code>__begin_seqs</code>	Begin of the sequence of iterator pairs.
<code>__end_seqs</code>	End of the sequence of iterator pairs.
<code>__rank</code>	The global rank to partition at.
<code>__begin_offsets</code>	A random-access <code>__sequence</code> <code>__begin</code> where the <code>__result</code> will be stored in. Each element of the sequence is an iterator that points to the first element on the greater part of the respective <code>__sequence</code> .
<code>__comp</code>	The ordering functor, defaults to <code>std::less<_Tp></code> .

Definition at line 122 of file `multiseq_selection.h`.

References `__rd_log2()`, `_GLIBCXX_CALL`, `std::vector<_Tp, _Alloc>::begin()`, `std::distance()`, `std::priority_queue<_Tp, _Sequence, _Compare>::empty()`, `std::vector<_Tp, _Alloc>::end()`, `std::make_pair()`, `std::max()`, `std::min()`, `std::priority_queue<_Tp, _Sequence, _Compare>::pop()`, `std::priority_queue<_Tp, _Sequence, _Compare>::push()`, `std::vector<_Tp, _Alloc>::push_back()`, and `std::priority_queue<_Tp, _Sequence, _Compare>::top()`.

Referenced by `multiway_merge_exact_splitting()`.

4.6.4.63 `template<typename _Tp , typename _RanSeqs , typename _RankType , typename _Compare > _Tp
__gnu_parallel::multiseq_selection (_RanSeqs __begin_seqs, _RanSeqs __end_seqs, _RankType __rank, _RankType &
__offset, _Compare __comp = std::less<_Tp>())`

Selects the element at a certain global `__rank` from several sorted sequences.

The sequences are passed via a sequence of random-access iterator pairs, none of the sequences may be empty.

Parameters

<code>__begin_seqs</code>	Begin of the sequence of iterator pairs.
<code>__end_seqs</code>	End of the sequence of iterator pairs.

Parameters

<code>__rank</code>	The global rank to partition at.
<code>__offset</code>	The rank of the selected element in the global subsequence of elements equal to the selected element. If the selected element is unique, this number is 0.
<code>__comp</code>	The ordering functor, defaults to <code>std::less</code> .

Definition at line 388 of file `multiseq_selection.h`.

References `__rd_log2()`, `__round_up_to_pow2()`, `_GLIBCXX_CALL`, `std::vector< _Tp, _Alloc >::begin()`, `std::distance()`, `std::priority_queue< _Tp, _Sequence, _Compare >::empty()`, `std::vector< _Tp, _Alloc >::end()`, `std::make_pair()`, `std::max()`, `std::min()`, `std::priority_queue< _Tp, _Sequence, _Compare >::pop()`, `std::priority_queue< _Tp, _Sequence, _Compare >::push()`, `std::vector< _Tp, _Alloc >::push_back()`, and `std::priority_queue< _Tp, _Sequence, _Compare >::top()`.

```
4.6.4.64 template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >
        _RAIterOut __gnu_parallel::multiway_merge ( _RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end,
        _RAIterOut __target, _DifferenceTp __length, _Compare __comp, __gnu_parallel::sequential_tag )
```

Multiway Merge Frontend.

Merge the sequences specified by `seqs_begin` and `__seqs_end` into `__target`. `__seqs_begin` and `__seqs_end` must point to a sequence of pairs. These pairs must contain an iterator to the beginning of a sequence in their first entry and an iterator the `_M_end` of the same sequence in their second entry.

Ties are broken arbitrarily. See `stable_multiway_merge` for a variant that breaks ties by sequence number but is slower.

The first entries of the pairs (i.e. the begin iterators) will be moved forward.

The output sequence has to provide enough space for all elements that are written to it.

This function will merge the input sequences:

- not stable
- parallel, depending on the input size and Settings
- using sampling for splitting
- not using sentinels

Example:

```
int sequences[10][10];
for (int __i = 0; __i < 10; ++__i)
    for (int __j = 0; __j < 10; ++__j)
        sequences[__i][__j] = __j;

int __out[33];
std::vector<std::pair<int*> > seqs;
for (int __i = 0; __i < 10; ++__i)
    { seqs.push(std::make_pair<int*>(sequences[__i],
                                    sequences[__i] + 10)) }

multiway_merge(seqs.begin(), seqs.end(), __target, std::less<int>(), 33);
```

See also

`stable_multiway_merge`

Precondition

All input sequences must be sorted.

Target must provide enough space to merge out length elements or the number of elements in all sequences, whichever is smaller.

Postcondition

[`__target`, return `__value`) contains merged `__elements` from the input sequences.

return `__value` - `__target` = min(`__length`, number of elements in all sequences).

Template Parameters

<code>_RAIterPairIterator</code>	iterator over sequence of pairs of iterators
<code>_RAIterOut</code>	iterator over target sequence
<code>_DifferenceTp</code>	difference type for the sequence
<code>_Compare</code>	strict weak ordering type to compare elements in sequences

Parameters

<code>__seqs_begin</code>	<code>__begin</code> of sequence <code>__sequence</code>
<code>__seqs_end</code>	<code>_M_end</code> of sequence <code>__sequence</code>
<code>__target</code>	target sequence to merge to.
<code>__comp</code>	strict weak ordering to use for element comparison.
<code>__length</code>	Maximum length to merge, possibly larger than the number of elements available.

Returns

`_M_end` iterator of output sequence

Definition at line 1418 of file `multiway_merge.h`.

References `__gnu_parallel::parallel_tag::__get_num_threads()`, `__sequential_multiway_merge()`, `_GLIBCXX_CALL`, `↔` `_GLIBCXX_PARALLEL_CONDITION`, `__gnu_parallel::Settings::get()`, `multiway_merge_exact_splitting()`, `multiway_↔` `merge_sampling_splitting()`, and `parallel_multiway_merge()`.

4.6.4.65 `template<template< typename RAI, typename C > class iterator, typename _RAIterIterator , typename _RAIter3 , typename _DifferenceTp , typename _Compare > _RAIter3 __gnu_parallel::multiway_merge_3_variant (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 __target, _DifferenceTp __length, _Compare __comp)`

Highly efficient 3-way merging procedure.

Merging is done with the algorithm implementation described by Peter Sanders. Basically, the idea is to minimize the number of necessary comparison after merging an element. The implementation trick that makes this fast is that the order of the sequences is stored in the instruction pointer (translated into labels in C++).

This works well for merging up to 4 sequences.

Note that making the merging stable does *not* come at a performance hit.

Whether the merging is done guarded or unguarded is selected by the used iterator class.

Parameters

<code>__seqs_begin</code>	Begin iterator of iterator pair input sequence.
<code>__seqs_end</code>	End iterator of iterator pair input sequence.
<code>__target</code>	Begin iterator of output sequence.
<code>__comp</code>	Comparator.
<code>__length</code>	Maximum length to merge, less equal than the total number of elements available.

Returns

End iterator of output sequence.

Definition at line 241 of file `multiway_merge.h`.

References `_GLIBCXX_CALL`.

4.6.4.66 `template<template< typename RAI, typename C > class iterator, typename _RAIterIterator , typename _RAIter3 , typename _DifferenceTp , typename _Compare > _RAIter3 __gnu_parallel::multiway_merge_4_variant (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 __target, _DifferenceTp __length, _Compare __comp)`

Highly efficient 4-way merging procedure.

Merging is done with the algorithm implementation described by Peter Sanders. Basically, the idea is to minimize the number of necessary comparison after merging an element. The implementation trick that makes this fast is that the order of the sequences is stored in the instruction pointer (translated into goto labels in C++).

This works well for merging up to 4 sequences.

Note that making the merging stable does *not* come at a performance hit.

Whether the merging is done guarded or unguarded is selected by the used iterator class.

Parameters

<code>__seqs_begin</code>	Begin iterator of iterator pair input sequence.
<code>__seqs_end</code>	End iterator of iterator pair input sequence.
<code>__target</code>	Begin iterator of output sequence.
<code>__comp</code>	Comparator.
<code>__length</code>	Maximum length to merge, less equal than the total number of elements available.

Returns

End iterator of output sequence.

Definition at line 360 of file `multiway_merge.h`.

References `_GLIBCXX_CALL`.

```
4.6.4.67 template<bool __stable, typename _RAIterlterator, typename _Compare, typename _DifferenceType > void
    __gnu_parallel::multiway_merge_exact_splitting ( _RAIterlterator __seqs_begin, _RAIterlterator __seqs_end,
    _DifferenceType __length, _DifferenceType __total_length, _Compare __comp, std::vector< std::pair<
    _DifferenceType, _DifferenceType > > * __pieces )
```

Exact splitting for parallel multiway-merge routine.

None of the passed sequences may be empty.

Definition at line 1120 of file `multiway_merge.h`.

References `__equally_split()`, `_GLIBCXX_PARALLEL_LENGTH`, `std::vector< _Tp, _Alloc >::begin()`, `std::vector< _Tp, _Alloc >::end()`, `multiseq_partition()`, and `std::vector< _Tp, _Alloc >::resize()`.

Referenced by `__parallel_merge_advance()`, `multiway_merge()`, and `multiway_merge_sentinels()`.

```
4.6.4.68 template<typename _LT, typename _RAIterlterator, typename _RAIter3, typename _DifferenceTp, typename _Compare >
    _RAIter3 __gnu_parallel::multiway_merge_loser_tree ( _RAIterlterator __seqs_begin, _RAIterlterator __seqs_end, _RAIter3
    __target, _DifferenceTp __length, _Compare __comp )
```

Multi-way merging procedure for a high branching factor, guarded case.

This merging variant uses a `LoserTree` class as selected by `_LT`.

Stability is selected through the used `LoserTree` class `_LT`.

At least one non-empty sequence is required.

Parameters

<code>__seqs_begin</code>	Begin iterator of iterator pair input sequence.
<code>__seqs_end</code>	End iterator of iterator pair input sequence.
<code>__target</code>	Begin iterator of output sequence.
<code>__comp</code>	Comparator.
<code>__length</code>	Maximum length to merge, less equal than the total number of elements available.

Returns

End iterator of output sequence.

Definition at line 491 of file `multiway_merge.h`.

References `_GLIBCXX_CALL`, and `_GLIBCXX_PARALLEL_LENGTH`.

```
4.6.4.69 template<typename UnguardedLoserTree , typename _RAIterIterator , typename _RAIter3 , typename _DifferenceTp ,
typename _Compare > _RAIter3 __gnu_parallel::multiway_merge_loser_tree_sentinel ( _RAIterIterator __seqs_begin,
_RAIterIterator __seqs_end, _RAIter3 __target, const typename std::iterator_traits< typename std::iterator_traits<
_RAIterIterator >::value_type::first_type >::value_type & __sentinel, _DifferenceTp __length, _Compare __comp )
```

Multi-way merging procedure for a high branching factor, requiring sentinels to exist.

Template Parameters

<i>UnguardedLoserTree</i>	<i>_Loser</i> Tree variant to use for the unguarded merging.
---------------------------	--

Parameters

<i>__seqs_begin</i>	Begin iterator of iterator pair input sequence.
<i>__seqs_end</i>	End iterator of iterator pair input sequence.
<i>__target</i>	Begin iterator of output sequence.
<i>__comp</i>	Comparator.
<i>__length</i>	Maximum length to merge, less equal than the total number of elements available.

Returns

End iterator of output sequence.

Definition at line 662 of file multiway_merge.h.

References `__is_sorted()`, and `_GLIBCXX_CALL`.

```
4.6.4.70 template<typename _LT , typename _RAIterIterator , typename _RAIter3 , typename _DifferenceTp , typename _Compare
> _RAIter3 __gnu_parallel::multiway_merge_loser_tree_unguarded ( _RAIterIterator __seqs_begin, _RAIterIterator
__seqs_end, _RAIter3 __target, const typename std::iterator_traits< typename std::iterator_traits< _RAIterIterator
>::value_type::first_type >::value_type & __sentinel, _DifferenceTp __length, _Compare __comp )
```

Multi-way merging procedure for a high branching factor, unguarded case.

Merging is done using the `LoserTree` class `_LT`.

Stability is selected by the used `LoserTrees`.

Precondition

No input will run out of elements during the merge.

Parameters

<i>__seqs_begin</i>	Begin iterator of iterator pair input sequence.
<i>__seqs_end</i>	End iterator of iterator pair input sequence.
<i>__target</i>	Begin iterator of output sequence.
<i>__comp</i>	Comparator.
<i>__length</i>	Maximum length to merge, less equal than the total number of elements available.

Returns

End iterator of output sequence.

Definition at line 574 of file `multiway_merge.h`.

References `_GLIBCXX_CALL`.

```
4.6.4.71 template<bool __stable, typename _RAIterlterator, typename _Compare, typename _DifferenceType > void
    __gnu_parallel::multiway_merge_sampling_splitting ( _RAIterlterator __seqs_begin, _RAIterlterator __seqs_end,
        _DifferenceType __length, _DifferenceType __total_length, _Compare __comp, std::vector< std::pair<
        _DifferenceType, _DifferenceType > > * __pieces )
```

Sampling based splitting for parallel multiway-merge routine.

Definition at line 1035 of file `multiway_merge.h`.

References `_GLIBCXX_PARALLEL_LENGTH`, `__gnu_parallel::_Settings::get()`, and `__gnu_parallel::_Settings::merge_oversampling`.

Referenced by `multiway_merge()`, and `multiway_merge_sentinels()`.

```
4.6.4.72 template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >
    _RAIterOut __gnu_parallel::multiway_merge_sentinels ( _RAIterPairlterator __seqs_begin, _RAIterPairlterator __seqs_end,
        _RAIterOut __target, _DifferenceTp __length, _Compare __comp, __gnu_parallel::sequential_tag )
```

Multiway Merge Frontend.

Merge the sequences specified by `seqs_begin` and `__seqs_end` into `__target`. `__seqs_begin` and `__seqs_end` must point to a sequence of pairs. These pairs must contain an iterator to the beginning of a sequence in their first entry and an iterator the `_M_end` of the same sequence in their second entry.

Ties are broken arbitrarily. See `stable_multiway_merge` for a variant that breaks ties by sequence number but is slower.

The first entries of the pairs (i.e. the begin iterators) will be moved forward accordingly.

The output sequence has to provide enough space for all elements that are written to it.

This function will merge the input sequences:

- not stable
- parallel, depending on the input size and Settings
- using sampling for splitting
- using sentinels

You have to take care that the element the `_M_end` iterator points to is readable and contains a value that is greater than any other non-sentinel value in all sequences.

Example:


```

int sequences[10][11];
for (int __i = 0; __i < 10; ++__i)
    for (int __j = 0; __i < 11; ++__j)
        sequences[__i][__j] = __j; // __last one is sentinel!

int __out[33];
std::vector<std::pair<int*> > seqs;
for (int __i = 0; __i < 10; ++__i)
    { seqs.push(std::make_pair<int*>(sequences[__i],
                                    sequences[__i] + 10)) }

multiway_merge(seqs.begin(), seqs.end(), __target, std::less<int>(), 33);

```

Precondition

All input sequences must be sorted.

Target must provide enough space to merge out length elements or the number of elements in all sequences, whichever is smaller.

For each `__i`, `__seqs_begin[__i].second` must be the end marker of the sequence, but also reference the one more `__sentinel` element.

Postcondition

[`__target`, return `__value`) contains merged `__elements` from the input sequences.
 return `__value` - `__target` = min(`__length`, number of elements in all sequences).

See also

`stable_multiway_merge_sentinels`

Template Parameters

<code>_RAIterPairIterator</code>	iterator over sequence of pairs of iterators
<code>_RAIterOut</code>	iterator over target sequence
<code>_DifferenceTp</code>	difference type for the sequence
<code>_Compare</code>	strict weak ordering type to compare elements in sequences

Parameters

<code>__seqs_begin</code>	<code>__begin</code> of sequence <code>__sequence</code>
<code>__seqs_end</code>	<code>_M_end</code> of sequence <code>__sequence</code>
<code>__target</code>	target sequence to merge to.
<code>__comp</code>	strict weak ordering to use for element comparison.
<code>__length</code>	Maximum length to merge, possibly larger than the number of elements available.

Returns

`_M_end` iterator of output sequence

Definition at line 1782 of file `multiway_merge.h`.

References `__gnu_parallel::parallel_tag::__get_num_threads()`, `__sequential_multiway_merge()`, `_GLIBCXX_CALL`, `↔` `_GLIBCXX_PARALLEL_CONDITION`, `__gnu_parallel::_Settings::get()`, `multiway_merge_exact_splitting()`, `multiway_↔` `merge_sampling_splitting()`, and `parallel_multiway_merge()`.

4.6.4.73 `template<bool __stable, bool __sentinels, typename _RAIter, typename _RAIter3, typename _DifferenceTp, typename _Splitter, typename _Compare> _RAIter3 __gnu_parallel::parallel_multiway_merge (_RAIter __seqs_begin, _RAIter __seqs_end, _RAIter3 __target, _Splitter __splitter, _DifferenceTp __length, _Compare __comp, _ThreadIndex __num_threads)`

Parallel multi-way merge routine.

The `_GLIBCXX_PARALLEL_DECISION` is based on the branching factor and runtime settings.

Must not be called if the number of sequences is 1.

Template Parameters

<code>_Splitter</code>	functor to split input (either <code>__exact</code> or sampling based)
<code>__stable</code>	Stable merging incurs a performance penalty.
<code>__sentinel</code>	Ignored.

Parameters

<code>__seqs_begin</code>	Begin iterator of iterator pair input sequence.
<code>__seqs_end</code>	End iterator of iterator pair input sequence.
<code>__target</code>	Begin iterator of output sequence.
<code>__comp</code>	Comparator.
<code>__length</code>	Maximum length to merge, possibly larger than the number of elements available.

Returns

End iterator of output sequence.

Definition at line 1225 of file `multiway_merge.h`.

References `__is_sorted()`, `_GLIBCXX_CALL`, `_GLIBCXX_PARALLEL_LENGTH`, `__gnu_parallel::_Settings::get()`, `std::↔` `make_pair()`, and `__gnu_parallel::_Settings::merge_oversampling`.

Referenced by `__parallel_merge_advance()`, `multiway_merge()`, and `multiway_merge_sentinels()`.

4.6.4.74 `template<bool __stable, bool __exact, typename _RAIter, typename _Compare> void __gnu_parallel::parallel_sort_mwms (_RAIter __begin, _RAIter __end, _Compare __comp, _ThreadIndex __num_threads)`

PMWMS main call.

Parameters

<code>__begin</code>	Begin iterator of sequence.
<code>__end</code>	End iterator of sequence.
<code>__comp</code>	Comparator.
<code>__num_threads</code>	Number of threads to use.

Definition at line 395 of file `multiway_mergesort.h`.

References `_GLIBCXX_CALL`, `__gnu_parallel::PMWMSSortingData<_RAIter>::M_num_threads`, `__gnu_parallel::PMWMSSortingData<_RAIter>::M_offsets`, `__gnu_parallel::PMWMSSortingData<_RAIter>::M_pieces`, `__gnu_parallel::PMWMSSortingData<_RAIter>::M_samples`, `__gnu_parallel::PMWMSSortingData<_RAIter>::M_source`, `__gnu_parallel::PMWMSSortingData<_RAIter>::M_starts`, `__gnu_parallel::PMWMSSortingData<_RAIter>::M_temporary`, `__gnu_parallel::Settings::get()`, and `__gnu_parallel::Settings::sort_mwms_oversampling`.

```
4.6.4.75 template<bool __stable, bool __exact, typename _RAIter, typename _Compare> void __gnu_parallel::parallel_sort_mwms_pu ( _PMWMSSortingData<_RAIter> * __sd, _Compare & __comp )
```

PMWMS code executed by each thread.

Parameters

<code>__sd</code>	Pointer to algorithm data.
<code>__comp</code>	Comparator.

Definition at line 308 of file `multiway_mergesort.h`.

References `__gnu_parallel::PMWMSSortingData<_RAIter>::M_num_threads`, `__gnu_parallel::PMWMSSortingData<_RAIter>::M_pieces`, `__gnu_parallel::PMWMSSortingData<_RAIter>::M_source`, `__gnu_parallel::PMWMSSortingData<_RAIter>::M_starts`, `__gnu_parallel::PMWMSSortingData<_RAIter>::M_temporary`, `__gnu_parallel::Settings::get()`, `std::make_pair()`, `__gnu_parallel::Settings::sort_mwms_oversampling`, and `std::uninitialized_copy()`.

4.6.5 Variable Documentation

```
4.6.5.1 const int __gnu_parallel::_CASable_bits [static]
```

Number of bits of `_CASable`.

Definition at line 130 of file `types.h`.

Referenced by `__decode2()`, and `__encode2()`.

4.6.5.2 `const __CASable __gnu_parallel::__CASable_mask` `[static]`

`__CASable` with the right half of bits set to 1.

Definition at line 133 of file `types.h`.

Referenced by `__decode2()`.

4.7 __gnu_pbds Namespace Reference

Classes

- struct [associative_tag](#)
- class [basic_branch](#)
- struct [basic_branch_tag](#)
- class [basic_hash_table](#)
- struct [basic_hash_tag](#)
- struct [basic_invalidation_guarantee](#)
- struct [binary_heap_tag](#)
- struct [binomial_heap_tag](#)
- class [cc_hash_max_collision_check_resize_trigger](#)
- class [cc_hash_table](#)
- struct [cc_hash_tag](#)
- struct [container_error](#)
- struct [container_tag](#)
- struct [container_traits](#)
- struct [container_traits_base](#)
- struct [container_traits_base< binary_heap_tag >](#)
- struct [container_traits_base< binomial_heap_tag >](#)
- struct [container_traits_base< cc_hash_tag >](#)
- struct [container_traits_base< gp_hash_tag >](#)
- struct [container_traits_base< list_update_tag >](#)
- struct [container_traits_base< ov_tree_tag >](#)
- struct [container_traits_base< pairing_heap_tag >](#)
- struct [container_traits_base< pat_trie_tag >](#)
- struct [container_traits_base< rb_tree_tag >](#)
- struct [container_traits_base< rc_binomial_heap_tag >](#)
- struct [container_traits_base< splay_tree_tag >](#)
- struct [container_traits_base< thin_heap_tag >](#)
- class [direct_mask_range_hashing](#)
- class [direct_mod_range_hashing](#)
- class [gp_hash_table](#)
- struct [gp_hash_tag](#)
- class [hash_exponential_size_policy](#)
- class [hash_load_check_resize_trigger](#)
- class [hash_prime_size_policy](#)
- class [hash_standard_resize_policy](#)
- struct [insert_error](#)
- struct [join_error](#)

- class [linear_probe_fn](#)
- class [list_update](#)
- struct [list_update_tag](#)
- class [lu_counter_policy](#)
- class [lu_move_to_front_policy](#)
- struct [null_node_update](#)
- struct [null_type](#)
- struct [ov_tree_tag](#)
- struct [pairing_heap_tag](#)
- struct [pat_trie_tag](#)
- struct [point_invalidation_guarantee](#)
- class [priority_queue](#)
- struct [priority_queue_tag](#)
- class [quadratic_probe_fn](#)
- struct [range_invalidation_guarantee](#)
- struct [rb_tree_tag](#)
- struct [rc_binomial_heap_tag](#)
- struct [resize_error](#)
- class [sample_probe_fn](#)
- class [sample_range_hashing](#)
- class [sample_ranged_hash_fn](#)
- class [sample_ranged_probe_fn](#)
- class [sample_resize_policy](#)
- class [sample_resize_trigger](#)
- class [sample_size_policy](#)
- class [sample_tree_node_update](#)
- struct [sample_trie_access_traits](#)
- class [sample_trie_node_update](#)
- struct [sample_update_policy](#)
- struct [sequence_tag](#)
- struct [splay_tree_tag](#)
- struct [string_tag](#)
- struct [thin_heap_tag](#)
- class [tree](#)
- class [tree_order_statistics_node_update](#)
- struct [tree_tag](#)
- class [trie](#)
- class [trie_order_statistics_node_update](#)
- class [trie_prefix_search_node_update](#)
- struct [trie_string_access_traits](#)
- struct [trie_tag](#)
- struct [trivial_iterator_tag](#)

Typedefs

- typedef void [trivial_iterator_difference_type](#)

Functions

- void `__throw_container_error()`
- void `__throw_insert_error()`
- void `__throw_join_error()`
- void `__throw_resize_error()`

4.7.1 Detailed Description

GNU extensions for policy-based data structures for public use.

4.8 `__gnu_profile` Namespace Reference

Classes

- class `__container_size_info`
- class `__container_size_stack_info`
- class `__hashfunc_info`
- class `__hashfunc_stack_info`
- class `__list2vector_info`
- class `__map2umap_info`
- class `__map2umap_stack_info`
- class `__object_info_base`
- struct `__reentrance_guard`
- class `__stack_hash`
- class `__trace_base`
- class `__trace_container_size`
- class `__trace_hash_func`
- class `__trace_hashtable_size`
- class `__trace_map2umap`
- class `__trace_vector_size`
- class `__trace_vector_to_list`
- class `__vector2list_info`
- class `__vector2list_stack_info`
- struct `__warning_data`

Typedefs

- typedef `std::vector< __cost_factor * >` `__cost_factor_vector`
- typedef `std::unordered_map< std::string, std::string >` `__env_t`
- typedef `void *` `__instruction_address_t`
- typedef `std::vector< __instruction_address_t >` `__stack_npt`
- typedef `__stack_npt *` `__stack_t`
- typedef `std::vector< __warning_data >` `__warning_vector_t`

Enumerations

- enum **__state_type** { **__ON**, **__OFF**, **__INVALID** }

Functions

- **std::size_t __env_to_size_t** (const char * __env_var, std::size_t __default_value)
- template<typename _InputIterator, typename _Function >
_Function **__for_each** (_InputIterator __first, _InputIterator __last, _Function __f)
- **__stack_t __get_stack** ()
- template<typename _Container >
void **__insert_top_n** (_Container & __output, const typename _Container::value_type & __value, typename _Container::size_type __n)
- bool **__is_invalid** ()
- bool **__is_off** ()
- bool **__is_on** ()
- int **__log2** (std::size_t __size)
- int **__log_magnitude** (float __f)
- float **__map_erase_cost** (std::size_t __size)
- float **__map_find_cost** (std::size_t __size)
- float **__map_insert_cost** (std::size_t __size)
- std::size_t **__max_mem** ()
- FILE * **__open_output_file** (const char * __extension)
- bool **__profcxx_init** ()
- void **__profcxx_init_unconditional** ()
- void **__read_cost_factors** ()
- template<typename _ForwardIterator, typename _Tp >
_ForwardIterator **__remove** (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __value)
- void **__report** ()
- void **__report_and_free** ()
- void **__set_cost_factors** ()
- void **__set_max_mem** ()
- void **__set_max_stack_trace_depth** ()
- void **__set_max_warn_count** ()
- void **__set_trace_path** ()
- std::size_t **__size** (__stack_t __stack)
- std::size_t **__stack_max_depth** ()
- template<typename _Container >
void **__top_n** (const _Container & __input, _Container & __output, typename _Container::size_type __n)
- **__hashfunc_info** * **__trace_hash_func_construct** ()
- void **__trace_hash_func_destruct** (**__hashfunc_info** *, std::size_t, std::size_t, std::size_t)
- void **__trace_hash_func_free** ()
- void **__trace_hash_func_init** ()
- void **__trace_hash_func_report** (FILE * __f, __warning_vector_t & __warnings)
- **__container_size_info** * **__trace_hashtable_size_construct** (std::size_t)
- void **__trace_hashtable_size_destruct** (**__container_size_info** *, std::size_t, std::size_t)
- void **__trace_hashtable_size_free** ()
- void **__trace_hashtable_size_init** ()
- void **__trace_hashtable_size_report** (FILE * __f, __warning_vector_t & __warnings)
- void **__trace_hashtable_size_resize** (**__container_size_info** *, std::size_t, std::size_t)

- `__list2slist_info * __trace_list_to_slist_construct ()`
- `void __trace_list_to_slist_destruct (__list2slist_info *)`
- `void __trace_list_to_slist_free ()`
- `void __trace_list_to_slist_init ()`
- `void __trace_list_to_slist_operation (__list2slist_info *)`
- `void __trace_list_to_slist_report (FILE * __f, __warning_vector_t & __warnings)`
- `void __trace_list_to_slist_rewind (__list2slist_info *)`
- `__list2vector_info * __trace_list_to_vector_construct ()`
- `void __trace_list_to_vector_destruct (__list2vector_info *)`
- `void __trace_list_to_vector_free ()`
- `void __trace_list_to_vector_init ()`
- `void __trace_list_to_vector_insert (__list2vector_info *, std::size_t, std::size_t)`
- `void __trace_list_to_vector_invalid_operator (__list2vector_info *)`
- `void __trace_list_to_vector_iterate (__list2vector_info *, int)`
- `void __trace_list_to_vector_report (FILE * __f, __warning_vector_t & __warnings)`
- `void __trace_list_to_vector_resize (__list2vector_info *, std::size_t, std::size_t)`
- `__map2umap_info * __trace_map_to_unordered_map_construct ()`
- `void __trace_map_to_unordered_map_destruct (__map2umap_info *)`
- `void __trace_map_to_unordered_map_erase (__map2umap_info *, std::size_t, std::size_t)`
- `void __trace_map_to_unordered_map_find (__map2umap_info *, std::size_t)`
- `void __trace_map_to_unordered_map_free ()`
- `void __trace_map_to_unordered_map_init ()`
- `void __trace_map_to_unordered_map_insert (__map2umap_info *, std::size_t, std::size_t)`
- `void __trace_map_to_unordered_map_invalidate (__map2umap_info *)`
- `void __trace_map_to_unordered_map_iterate (__map2umap_info *, std::size_t)`
- `void __trace_map_to_unordered_map_iterate (__map2umap_info * __info, int)`
- `void __trace_map_to_unordered_map_report (FILE * __f, __warning_vector_t & __warnings)`
- `template<typename __object_info, typename __stack_info >`
`void __trace_report (__trace_base< __object_info, __stack_info > * __cont, FILE * __f, __warning_vector_t & __warnings)`
- `__container_size_info * __trace_vector_size_construct (std::size_t)`
- `void __trace_vector_size_destruct (__container_size_info *, std::size_t, std::size_t)`
- `void __trace_vector_size_free ()`
- `void __trace_vector_size_init ()`
- `void __trace_vector_size_report (FILE *, __warning_vector_t &)`
- `void __trace_vector_size_resize (__container_size_info *, std::size_t, std::size_t)`
- `__vector2list_info * __trace_vector_to_list_construct ()`
- `void __trace_vector_to_list_destruct (__vector2list_info *)`
- `void __trace_vector_to_list_free ()`
- `void __trace_vector_to_list_init ()`
- `void __trace_vector_to_list_insert (__vector2list_info *, std::size_t, std::size_t)`
- `void __trace_vector_to_list_invalid_operator (__vector2list_info *)`
- `void __trace_vector_to_list_iterate (__vector2list_info *, int)`
- `void __trace_vector_to_list_report (FILE *, __warning_vector_t &)`
- `void __trace_vector_to_list_resize (__vector2list_info *, std::size_t, std::size_t)`
- `bool __turn (__state_type __s)`
- `bool __turn_off ()`
- `bool __turn_on ()`
- `void __write (FILE * __f, __stack_t __stack)`
- `void __write_cost_factors ()`
- `_GLIBCXX_PROFILE_DEFINE_DATA (__state_type, __state, __INVALID)`

- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__trace_hash_func` *, `_S_hash_func`, 0)
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__trace_hashtable_size` *, `_S_hashtable_size`, 0)
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__trace_map2umap` *, `_S_map2umap`, 0)
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__trace_vector_size` *, `_S_vector_size`, 0)
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__trace_vector_to_list` *, `_S_vector_to_list`, 0)
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__trace_list_to_slist` *, `_S_list_to_slist`, 0)
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__trace_list_to_vector` *, `_S_list_to_vector`, 0)
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__cost_factor`, `__vector_shift_cost_factor`, {"__vector_shift_cost_factor", 1.0})
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__cost_factor`, `__vector_iterate_cost_factor`, {"__vector_iterate_cost_factor", 1.0})
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__cost_factor`, `__vector_resize_cost_factor`, {"__vector_resize_cost_factor", 1.0})
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__cost_factor`, `__list_shift_cost_factor`, {"__list_shift_cost_factor", 0.0})
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__cost_factor`, `__list_iterate_cost_factor`, {"__list_iterate_cost_factor", 10.0})
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__cost_factor`, `__list_resize_cost_factor`, {"__list_resize_cost_factor", 0.0})
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__cost_factor`, `__map_insert_cost_factor`, {"__map_insert_cost_factor", 1.5})
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__cost_factor`, `__map_erase_cost_factor`, {"__map_erase_cost_factor", 1.5})
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__cost_factor`, `__map_find_cost_factor`, {"__map_find_cost_factor", 1})
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__cost_factor`, `__map_iterate_cost_factor`, {"__map_iterate_cost_factor", 2.3})
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__cost_factor`, `__umap_insert_cost_factor`, {"__umap_insert_cost_factor", 12.0})
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__cost_factor`, `__umap_erase_cost_factor`, {"__umap_erase_cost_factor", 12.0})
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__cost_factor`, `__umap_find_cost_factor`, {"__umap_find_cost_factor", 10.0})
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__cost_factor`, `__umap_iterate_cost_factor`, {"__umap_iterate_cost_factor", 1.7})
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`__cost_factor_vector` *, `__cost_factors`, 0)
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`const char` *, `_S_trace_file_name`, `_GLIBCXX_PROFILE_TRACE_PATH_ROOT`)
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`std::size_t`, `_S_max_warn_count`, `_GLIBCXX_PROFILE_MAX_WARN_COUNT`)
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`std::size_t`, `_S_max_stack_depth`, `_GLIBCXX_PROFILE_MAX_STACK_DEPTH`)
- `_GLIBCXX_PROFILE_DEFINE_DATA` (`std::size_t`, `_S_max_mem`, `_GLIBCXX_PROFILE_MEM_PER_DIAGNOSTIC`)
- `_GLIBCXX_PROFILE_DEFINE_UNINIT_DATA` (`__env_t`, `__env`)
- `_GLIBCXX_PROFILE_DEFINE_UNINIT_DATA` (`__gnu_cxx::__mutex`, `__global_mutex`)

4.8.1 Detailed Description

GNU profile code for public use.

4.8.2 Typedef Documentation

4.8.2.1 `typedef std::unordered_map<std::string, std::string> __gnu_profile::__env_t`

Internal environment. Values can be set one of two ways: 1. In config file "var = value". The default config file path is `libstdc++-profile.conf`. 2. By setting process environment variables. For instance, in a Bash shell you can set the unit cost of iterating through a map like this: `export __map_iterate_cost_factor=5.0`. If a value is set both in the input file and through an environment variable, the environment value takes precedence.

Definition at line 65 of file `profiler_trace.h`.

4.8.3 Function Documentation

4.8.3.1 `bool __gnu_profile::__profcxx_init() [inline]`

This function must be called by each instrumentation point.

The common path is inlined fully.

Definition at line 653 of file `profiler_trace.h`.

4.8.3.2 `void __gnu_profile::__report() [inline]`

Final report method, registered with **atexit**.

This can also be called directly by user code, including signal handlers. It is protected against deadlocks by the reentrance guard in `profiler.h`. However, when called from a signal handler that triggers while within `__gnu_profile` (under the guarded zone), no output will be produced.

Definition at line 448 of file `profiler_trace.h`.

References `std::begin()`, `std::basic_string<_CharT, _Traits, _Alloc>::begin()`, `std::basic_string<_CharT, _Traits, _Alloc>::c_str()`, `std::end()`, `std::basic_string<_CharT, _Traits, _Alloc>::end()`, `std::basic_string<_CharT, _Traits, _Alloc>::erase()`, `std::basic_string<_CharT, _Traits, _Alloc>::find()`, `std::basic_string<_CharT, _Traits, _Alloc>::find_first_of()`, `std::basic_string<_CharT, _Traits, _Alloc>::find_first_not_of()`, `std::getline()`, `std::basic_string<_CharT, _Traits, _Alloc>::length()`, `std::min()`, and `std::basic_string<_CharT, _Traits, _Alloc>::substr()`.

4.8.3.3 `__gnu_profile::__GLIBCXX_PROFILE_DEFINE_UNINIT_DATA(__gnu_cxx::__mutex, __global_mutex)`

Master lock.

4.9 `__gnu_sequential` Namespace Reference

4.9.1 Detailed Description

GNU sequential classes for public use.

4.10 abi Namespace Reference

4.10.1 Detailed Description

The cross-vendor C++ Application Binary Interface. A namespace alias to `__cxxabiv1`, but user programs should use the alias 'abi'.

A brief overview of an ABI is given in the libstdc++ FAQ, question 5.8 (you may have a copy of the FAQ locally, or you can view the online version at http://gcc.gnu.org/onlinedocs/libstdc++/faq.html#5_8).

GCC subscribes to a cross-vendor ABI for C++, sometimes called the IA64 ABI because it happens to be the native ABI for that platform. It is summarized at <http://www.codesourcery.com/cxx-abi/> along with the current specification.

For users of GCC greater than or equal to 3.x, entry points are available in `<cxxabi.h>`, which notes, *'It is not normally necessary for user programs to include this header, or use the entry points directly. However, this header is available should that be needed.'*

4.11 std Namespace Reference

Namespaces

- [__debug](#)
- [__detail](#)
- [__parallel](#)
- [__profile](#)
- [chrono](#)
- [decimal](#)
- [placeholders](#)
- [regex_constants](#)
- [rel_ops](#)
- [this_thread](#)
- [tr1](#)
- [tr2](#)

Classes

- [struct __allocated_ptr](#)
- [struct __atomic_base](#)
- [struct __atomic_base< _PTp * >](#)
- [struct __atomic_flag_base](#)
- [class __basic_future](#)
- [class __codecvt_abstract_base](#)
- [class __ctype_abstract_base](#)
- [struct __future_base](#)
- [struct __is_location_invariant](#)
- [struct __is_nullptr_t](#)
- [struct __is_tuple_like_impl< std::pair< _T1, _T2 > >](#)
- [struct __iterator_traits](#)

- struct [__numeric_limits_base](#)
- class [__shared_mutex_cv](#)
- struct [_Base_bitset](#)
- struct [_Base_bitset< 0 >](#)
- struct [_Base_bitset< 1 >](#)
- struct [_Bind](#)
- struct [_Bind_result](#)
- class [_Deque_base](#)
- struct [_Deque_iterator](#)
- struct [_Enable_copy_move](#)
- struct [_Enable_default_constructor](#)
- struct [_Enable_destructor](#)
- struct [_Enable_special_members](#)
- class [_Function_base](#)
- struct [_Fwd_list_base](#)
- struct [_Fwd_list_const_iterator](#)
- struct [_Fwd_list_iterator](#)
- struct [_Fwd_list_node](#)
- struct [_Fwd_list_node_base](#)
- class [_Hashtable](#)
- class [_List_base](#)
- struct [_List_const_iterator](#)
- struct [_List_iterator](#)
- struct [_List_node](#)
- struct [_Maybe_get_result_type](#)
- struct [_Maybe_unary_or_binary_function](#)
- struct [_Maybe_unary_or_binary_function< _Res, _T1 >](#)
- struct [_Maybe_unary_or_binary_function< _Res, _T1, _T2 >](#)
- struct [_Maybe_wrap_member_pointer](#)
- struct [_Maybe_wrap_member_pointer< _Tp _Class::* >](#)
- class [_Mu](#)
- class [_Mu< _Arg, false, false >](#)
- class [_Mu< _Arg, false, true >](#)
- class [_Mu< _Arg, true, false >](#)
- class [_Mu< reference_wrapper< _Tp >, false, false >](#)
- struct [_Placeholder](#)
- struct [_Reference_wrapper_base](#)
- struct [_Reference_wrapper_base_impl](#)
- struct [_Sp_ebo_helper< _Nm, _Tp, false >](#)
- struct [_Sp_ebo_helper< _Nm, _Tp, true >](#)
- class [_Temporary_buffer](#)
- struct [_Tuple_impl](#)
- struct [_Tuple_impl< _Idx, _Head, _Tail... >](#)
- struct [_Vector_base](#)
- struct [_Weak_result_type](#)
- struct [_Weak_result_type_impl](#)
- struct [_Weak_result_type_impl< _Res\(&\)\(_ArgTypes...\)>](#)
- struct [_Weak_result_type_impl< _Res\(*\)\(_ArgTypes...\)>](#)
- struct [_Weak_result_type_impl< _Res\(_ArgTypes...\)>](#)
- struct [_Weak_result_type_impl< _Res\(_Class::*\)\(_ArgTypes...\) const >](#)
- struct [_Weak_result_type_impl< _Res\(_Class::*\)\(_ArgTypes...\) const volatile >](#)

- struct [_Weak_result_type_impl< _Res\(_Class::*\)\(_ArgTypes...\) volatile >](#)
- struct [_Weak_result_type_impl< _Res\(_Class::*\)\(_ArgTypes...\) >](#)
- struct [adopt_lock_t](#)
- class [allocator](#)
- class [allocator< void >](#)
- struct [allocator_arg_t](#)
- struct [allocator_traits](#)
- struct [allocator_traits< allocator< _Tp > >](#)
- struct [array](#)
- struct [atomic](#)
- struct [atomic< _Tp * >](#)
- struct [atomic< bool >](#)
- struct [atomic< char >](#)
- struct [atomic< char16_t >](#)
- struct [atomic< char32_t >](#)
- struct [atomic< int >](#)
- struct [atomic< long >](#)
- struct [atomic< long long >](#)
- struct [atomic< short >](#)
- struct [atomic< signed char >](#)
- struct [atomic< unsigned char >](#)
- struct [atomic< unsigned int >](#)
- struct [atomic< unsigned long >](#)
- struct [atomic< unsigned long long >](#)
- struct [atomic< unsigned short >](#)
- struct [atomic< wchar_t >](#)
- struct [atomic_flag](#)
- class [auto_ptr](#)
- struct [auto_ptr_ref](#)
- class [back_insert_iterator](#)
- class [bad_alloc](#)
- class [bad_cast](#)
- class [bad_exception](#)
- class [bad_function_call](#)
- class [bad_typeid](#)
- class [bad_weak_ptr](#)
- class [basic_filebuf](#)
- class [basic_fstream](#)
- class [basic_ifstream](#)
- class [basic_ios](#)
- class [basic_iostream](#)
- class [basic_istream](#)
- class [basic_istreambuf_iterator](#)
- class [basic_ofstream](#)
- class [basic_ostream](#)
- class [basic_ostreambuf_iterator](#)
- class [basic_regex](#)
- class [basic_streambuf](#)
- class [basic_string](#)
- class [basic_stringbuf](#)
- class [basic_stringstream](#)

- class [bernoulli_distribution](#)
- struct [bidirectional_iterator_tag](#)
- struct [binary_function](#)
- class [binary_negate](#)
- class [binder1st](#)
- class [binder2nd](#)
- class [binomial_distribution](#)
- class [bitset](#)
- class [cauchy_distribution](#)
- struct [char_traits](#)
- struct [char_traits< __gnu_cxx::character< _Value, _Int, _St > >](#)
- struct [char_traits< char >](#)
- struct [char_traits< wchar_t >](#)
- class [chi_squared_distribution](#)
- class [codecvt](#)
- class [codecvt< _InternT, _ExternT, encoding_state >](#)
- class [codecvt< char, char, mbstate_t >](#)
- class [codecvt< char16_t, char, mbstate_t >](#)
- class [codecvt< char32_t, char, mbstate_t >](#)
- class [codecvt< wchar_t, char, mbstate_t >](#)
- class [codecvt_base](#)
- class [codecvt_byname](#)
- class [collate](#)
- class [collate_byname](#)
- struct [complex](#)
- struct [complex< double >](#)
- struct [complex< float >](#)
- struct [complex< long double >](#)
- class [condition_variable](#)
- class [const_mem_fun1_ref_t](#)
- class [const_mem_fun1_t](#)
- class [const_mem_fun_ref_t](#)
- class [const_mem_fun_t](#)
- class [ctype](#)
- class [ctype< char >](#)
- class [ctype< wchar_t >](#)
- struct [ctype_base](#)
- class [ctype_byname](#)
- class [ctype_byname< char >](#)
- struct [default_delete](#)
- struct [default_delete< _Tp\[\]>](#)
- struct [defer_lock_t](#)
- class [deque](#)
- class [discard_block_engine](#)
- class [discrete_distribution](#)
- struct [divides](#)
- struct [divides< void >](#)
- class [domain_error](#)
- class [enable_shared_from_this](#)
- struct [equal_to](#)
- struct [equal_to< void >](#)

- struct `error_code`
- struct `error_condition`
- class `exception`
- class `exponential_distribution`
- class `extreme_value_distribution`
- class `fisher_f_distribution`
- struct `forward_iterator_tag`
- class `forward_list`
- class `fpos`
- class `front_insert_iterator`
- class `function< _Res(_ArgTypes...)>`
- class `future`
- class `future< _Res & >`
- class `future< void >`
- class `future_error`
- class `gamma_distribution`
- class `geometric_distribution`
- struct `greater`
- struct `greater< void >`
- struct `greater_equal`
- struct `greater_equal< void >`
- class `gslice`
- class `gslice_array`
- struct `hash`
- struct `hash< __debug::bitset< _Nb > >`
- struct `hash< __debug::vector< bool, _Alloc > >`
- struct `hash< __gnu_cxx::__u16vstring >`
- struct `hash< __gnu_cxx::__u32vstring >`
- struct `hash< __gnu_cxx::__vstring >`
- struct `hash< __gnu_cxx::__wvstring >`
- struct `hash< __gnu_cxx::throw_value_limit >`
- struct `hash< __gnu_cxx::throw_value_random >`
- struct `hash< __profile::bitset< _Nb > >`
- struct `hash< __profile::vector< bool, _Alloc > >`
- struct `hash< __shared_ptr< _Tp, _Lp > >`
- struct `hash< _Tp * >`
- struct `hash< bool >`
- struct `hash< char >`
- struct `hash< char16_t >`
- struct `hash< char32_t >`
- struct `hash< double >`
- struct `hash< error_code >`
- struct `hash< experimental::shared_ptr< _Tp > >`
- struct `hash< float >`
- struct `hash< int >`
- struct `hash< long >`
- struct `hash< long double >`
- struct `hash< long long >`
- struct `hash< shared_ptr< _Tp > >`
- struct `hash< short >`
- struct `hash< signed char >`

- struct [hash< string >](#)
- struct [hash< thread::id >](#)
- struct [hash< type_index >](#)
- struct [hash< u16string >](#)
- struct [hash< u32string >](#)
- struct [hash< unique_ptr< _Tp, _Dp > >](#)
- struct [hash< unsigned char >](#)
- struct [hash< unsigned int >](#)
- struct [hash< unsigned long >](#)
- struct [hash< unsigned long long >](#)
- struct [hash< unsigned short >](#)
- struct [hash< wchar_t >](#)
- struct [hash< wstring >](#)
- struct [hash<::bitset< _Nb > >](#)
- struct [hash<::vector< bool, _Alloc > >](#)
- class [independent_bits_engine](#)
- class [indirect_array](#)
- class [initializer_list](#)
- struct [input_iterator_tag](#)
- class [insert_iterator](#)
- struct [integer_sequence](#)
- struct [integral_constant](#)
- class [invalid_argument](#)
- class [ios_base](#)
- struct [is_abstract](#)
- struct [is_arithmetic](#)
- struct [is_array](#)
- struct [is_bind_expression](#)
- struct [is_bind_expression< _Bind< _Signature > >](#)
- struct [is_bind_expression< _Bind_result< _Result, _Signature > >](#)
- struct [is_bind_expression< const _Bind< _Signature > >](#)
- struct [is_bind_expression< const _Bind_result< _Result, _Signature > >](#)
- struct [is_bind_expression< const volatile _Bind< _Signature > >](#)
- struct [is_bind_expression< const volatile _Bind_result< _Result, _Signature > >](#)
- struct [is_bind_expression< volatile _Bind< _Signature > >](#)
- struct [is_bind_expression< volatile _Bind_result< _Result, _Signature > >](#)
- struct [is_class](#)
- struct [is_compound](#)
- struct [is_const](#)
- struct [is_empty](#)
- struct [is_enum](#)
- struct [is_error_code_enum](#)
- struct [is_error_code_enum< future_errc >](#)
- struct [is_error_condition_enum](#)
- struct [is_final](#)
- struct [is_floating_point](#)
- struct [is_function](#)
- struct [is_fundamental](#)
- struct [is_integral](#)
- struct [is_literal_type](#)
- struct [is_lvalue_reference](#)

- struct [is_member_function_pointer](#)
- struct [is_member_object_pointer](#)
- struct [is_member_pointer](#)
- struct [is_null_pointer](#)
- struct [is_object](#)
- struct [is_placeholder](#)
- struct [is_placeholder< _Placeholder< _Num > >](#)
- struct [is_pod](#)
- struct [is_pointer](#)
- struct [is_polymorphic](#)
- struct [is_reference](#)
- struct [is_rvalue_reference](#)
- struct [is_scalar](#)
- struct [is_standard_layout](#)
- struct [is_trivial](#)
- struct [is_union](#)
- struct [is_void](#)
- struct [is_volatile](#)
- class [istream_iterator](#)
- class [istreambuf_iterator](#)
- struct [iterator](#)
- struct [iterator_traits< _Tp * >](#)
- struct [iterator_traits< const _Tp * >](#)
- class [length_error](#)
- struct [less](#)
- struct [less< void >](#)
- struct [less_equal](#)
- struct [less_equal< void >](#)
- class [linear_congruential_engine](#)
- class [list](#)
- class [locale](#)
- class [lock_guard](#)
- class [logic_error](#)
- struct [logical_and](#)
- struct [logical_and< void >](#)
- struct [logical_not](#)
- struct [logical_not< void >](#)
- struct [logical_or](#)
- struct [logical_or< void >](#)
- class [lognormal_distribution](#)
- class [map](#)
- class [mask_array](#)
- class [match_results](#)
- class [mem_fun1_ref_t](#)
- class [mem_fun1_t](#)
- class [mem_fun_ref_t](#)
- class [mem_fun_t](#)
- class [mersenne_twister_engine](#)
- class [messages](#)
- struct [messages_base](#)
- class [messages_byname](#)

- struct [minus](#)
- struct [minus< void >](#)
- struct [modulus](#)
- struct [modulus< void >](#)
- class [money_base](#)
- class [money_get](#)
- class [money_put](#)
- class [moneypunct](#)
- class [moneypunct_byname](#)
- class [move_iterator](#)
- class [multimap](#)
- struct [multiplies](#)
- struct [multiplies< void >](#)
- class [multiset](#)
- class [mutex](#)
- struct [negate](#)
- struct [negate< void >](#)
- class [negative_binomial_distribution](#)
- class [nested_exception](#)
- class [normal_distribution](#)
- struct [not_equal_to](#)
- struct [not_equal_to< void >](#)
- class [num_get](#)
- class [num_put](#)
- struct [numeric_limits](#)
- struct [numeric_limits< bool >](#)
- struct [numeric_limits< char >](#)
- struct [numeric_limits< char16_t >](#)
- struct [numeric_limits< char32_t >](#)
- struct [numeric_limits< double >](#)
- struct [numeric_limits< float >](#)
- struct [numeric_limits< int >](#)
- struct [numeric_limits< long >](#)
- struct [numeric_limits< long double >](#)
- struct [numeric_limits< long long >](#)
- struct [numeric_limits< short >](#)
- struct [numeric_limits< signed char >](#)
- struct [numeric_limits< unsigned char >](#)
- struct [numeric_limits< unsigned int >](#)
- struct [numeric_limits< unsigned long >](#)
- struct [numeric_limits< unsigned long long >](#)
- struct [numeric_limits< unsigned short >](#)
- struct [numeric_limits< wchar_t >](#)
- class [numpunct](#)
- class [numpunct_byname](#)
- struct [once_flag](#)
- class [ostream_iterator](#)
- class [ostreambuf_iterator](#)
- class [out_of_range](#)
- struct [output_iterator_tag](#)
- class [overflow_error](#)

- struct [owner_less](#)
- struct [owner_less< shared_ptr< _Tp > >](#)
- struct [owner_less< weak_ptr< _Tp > >](#)
- class [packaged_task< _Res\(_ArgTypes...\)>](#)
- struct [pair](#)
- class [piecewise_constant_distribution](#)
- struct [piecewise_construct_t](#)
- class [piecewise_linear_distribution](#)
- struct [plus](#)
- class [pointer_to_binary_function](#)
- class [pointer_to_unary_function](#)
- struct [pointer_traits](#)
- struct [pointer_traits< _Tp * >](#)
- class [poisson_distribution](#)
- class [priority_queue](#)
- class [promise](#)
- class [promise< _Res & >](#)
- class [promise< void >](#)
- class [queue](#)
- struct [random_access_iterator_tag](#)
- class [random_device](#)
- class [range_error](#)
- struct [ratio](#)
- struct [ratio_equal](#)
- struct [ratio_not_equal](#)
- class [raw_storage_iterator](#)
- class [recursive_mutex](#)
- class [recursive_timed_mutex](#)
- class [reference_wrapper](#)
- class [regex_error](#)
- class [regex_iterator](#)
- class [regex_token_iterator](#)
- struct [regex_traits](#)
- class [reverse_iterator](#)
- class [runtime_error](#)
- class [scoped_allocator_adaptor](#)
- class [seed_seq](#)
- class [set](#)
- class [shared_future](#)
- class [shared_future< _Res & >](#)
- class [shared_future< void >](#)
- class [shared_lock](#)
- class [shared_ptr](#)
- class [shared_timed_mutex](#)
- class [shuffle_order_engine](#)
- class [slice](#)
- class [slice_array](#)
- class [stack](#)
- class [student_t_distribution](#)
- class [sub_match](#)
- class [subtract_with_carry_engine](#)

- class `system_error`
- class `thread`
- class `time_base`
- class `time_get`
- class `time_get_byname`
- class `time_put`
- class `time_put_byname`
- class `timed_mutex`
- struct `try_to_lock_t`
- class `tuple`
- class `tuple< _T1, _T2 >`
- struct `tuple_element`
- struct `tuple_element< 0, std::pair< _Tp1, _Tp2 > >`
- struct `tuple_element< 0, tuple< _Head, _Tail... > >`
- struct `tuple_element< 1, std::pair< _Tp1, _Tp2 > >`
- struct `tuple_element< __i, tuple< _Head, _Tail... > >`
- struct `tuple_element< _Int, std::__debug::array< _Tp, _Nm > >`
- struct `tuple_element< _Int,::array< _Tp, _Nm > >`
- struct `tuple_size`
- struct `tuple_size< std::__debug::array< _Tp, _Nm > >`
- struct `tuple_size< std::pair< _Tp1, _Tp2 > >`
- struct `tuple_size< tuple< _Elements... > >`
- struct `tuple_size<::array< _Tp, _Nm > >`
- struct `type_index`
- class `type_info`
- struct `unary_function`
- class `unary_negate`
- class `underflow_error`
- class `uniform_int_distribution`
- class `uniform_real_distribution`
- class `unique_lock`
- class `unique_ptr`
- class `unique_ptr< _Tp[], _Dp >`
- class `unordered_map`
- class `unordered_multimap`
- class `unordered_multiset`
- class `unordered_set`
- struct `uses_allocator`
- struct `uses_allocator< tuple< _Types... >, _Alloc >`
- class `valarray`
- class `vector`
- class `vector< bool, _Alloc >`
- class `wbuffer_convert`
- class `weak_ptr`
- class `weibull_distribution`
- class `wstring_convert`

Typedefs

- `template<typename _Alloc , typename _Up >`
`using __alloc_rebind = __detected_or_t< __replace_first_arg_t, __allocator_traits_base::__rebind, _Alloc, ↵`
`_Up >`
- `template<typename _Tp >`
`using __allocator_base = __gnu_cxx::new_allocator< _Tp >`
- `template<typename _Fn , typename... _Args>`
`using __async_result_of = typename result_of< typename decay< _Fn >::type(typename decay< _Args >↵`
`::type...)>::type`
- `typedef unsigned char __atomic_flag_data_type`
- `template<bool __v>`
`using __bool_constant = integral_constant< bool, __v >`
- `typedef FILE __c_file`
- `typedef __locale_t __c_locale`
- `typedef __gthread_mutex_t __c_lock`
- `template<typename _Tp , typename _Hash >`
`using __cache_default = __not< __and< __is_fast_hash< _Hash >, __detail::__is_noexcept_hash< _Tp,`
`_Hash >>>`
- `template<typename _From , typename _To >`
`using __check_func_return_type = __or< is_void< _To >, is_same< _From, _To >, is_convertible< _From,`
`_To >>`
- `typedef basic_string< char > __cow_string`
- `template<typename _Tp >`
`using __empty_not_final = typename conditional< __is_final(_Tp), false_type, __is_empty_non_tuple< _Tp`
`>>::type`
- `template<typename _Tp >`
`using __get_first_arg_t = typename __get_first_arg< _Tp >::type`
- `template<typename _Alloc , typename _Tp >`
`using __is_erased_or_convertible = __or< is_same< _Tp, __erased_type >, is_convertible< _Alloc, _Tp`
`>>`
- `template<typename _Tp , typename _Tp2 = typename decay< _Tp>::type>`
`using __is_socketlike = __or< is_integral< _Tp2 >, is_enum< _Tp2 >>`
- `template<typename _Tp >`
`using __make_not_void = typename conditional< is_void< _Tp >::value, __undefined, _Tp >::type`
- `template<typename _Alloc >`
`using __outer_allocator_t = decltype(std::declval< _Alloc >().outer_allocator())`
- `template<typename _Ptr , typename _Tp >`
`using __ptr_rebind = typename pointer_traits< _Ptr >::template rebind< _Tp >`
- `template<typename _Tp , typename _Up >`
`using __replace_first_arg_t = typename __replace_first_arg< _Tp, _Up >::type`
- `template<typename _Tp >`
`using __rethrow_if_nested_cond = typename enable_if< __and< is_polymorphic< _Tp >, __or< __not<`
`is_base_of< nested_exception, _Tp >>, is_convertible< _Tp *, nested_exception * >>>::value >::type`
- `typedef basic_string< char > __sso_string`
- `template<typename _Bi_iter , typename _Ch_traits , typename _Ch_alloc >`
`using __sub_match_string = basic_string< typename iterator_traits< _Bi_iter >::value_type, _Ch_traits, _Ch↵`
`_alloc >`
- `template<std::size_t __i, typename _Tp >`
`using __tuple_element_t = typename tuple_element< __i, _Tp >::type`

- `template<typename _Key, typename _Tp, typename _Hash = hash<_Key>, typename _Pred = std::equal_to<_Key>, typename _Alloc = std::allocator<std::pair<const _Key, _Tp> >, typename _Tr = __umap_traits<__cache_default<_Key, _Hash>::value>>
using __umap_hashtable = __Hashtable<_Key, std::pair< const _Key, _Tp >, _Alloc, __detail::__Select1st, _↵
_Pred, _Hash, __detail::__Mod_range_hashing, __detail::__Default_ranged_hash, __detail::__Prime_rehash_policy,
_Tr >`
- `template<bool _Cache>
using __umap_traits = __detail::__Hashtable_traits<_Cache, false, true >`
- `template<typename _Key, typename _Tp, typename _Hash = hash<_Key>, typename _Pred = std::equal_to<_Key>, typename _Alloc = std::allocator<std::pair<const _Key, _Tp> >, typename _Tr = __umap_traits<__cache_default<_Key, _Hash>::value>>
using __ummap_hashtable = __Hashtable<_Key, std::pair< const _Key, _Tp >, _Alloc, __detail::__Select1st, _↵
_Pred, _Hash, __detail::__Mod_range_hashing, __detail::__Default_ranged_hash, __detail::__Prime_rehash_policy,
_Tr >`
- `template<bool _Cache>
using __ummap_traits = __detail::__Hashtable_traits<_Cache, false, false >`
- `template<typename _Value, typename _Hash = hash<_Value>, typename _Pred = std::equal_to<_Value>, typename _Alloc = std::↵
::allocator<_Value>, typename _Tr = __umset_traits<__cache_default<_Value, _Hash>::value>>
using __umset_hashtable = __Hashtable<_Value, _Value, _Alloc, __detail::__Identity, _Pred, _Hash, __detail::__↵
_Mod_range_hashing, __detail::__Default_ranged_hash, __detail::__Prime_rehash_policy, _Tr >`
- `template<bool _Cache>
using __umset_traits = __detail::__Hashtable_traits<_Cache, true, false >`
- `template<typename _Tp, typename _Alloc, typename... _Args>
using __uses_alloc_t = __uses_alloc< uses_allocator<_Tp, _Alloc >::value, _Tp, _Alloc, _Args... >`
- `template<typename _Value, typename _Hash = hash<_Value>, typename _Pred = std::equal_to<_Value>, typename _Alloc = std::↵
::allocator<_Value>, typename _Tr = __uset_traits<__cache_default<_Value, _Hash>::value>>
using __uset_hashtable = __Hashtable<_Value, _Value, _Alloc, __detail::__Identity, _Pred, _Hash, __detail::__↵
_Mod_range_hashing, __detail::__Default_ranged_hash, __detail::__Prime_rehash_policy, _Tr >`
- `template<bool _Cache>
using __uset_traits = __detail::__Hashtable_traits<_Cache, true, true >`
- `typedef unsigned long _Bit_type`
- `template<typename _Tp1, typename _Tp2 >
using _NotSame = __not< is_same< typename std::decay<_Tp1 >::type, typename std::decay<_Tp2 >↵
::type >>`
- `template<typename _InIter >
using _RequireInputIter = typename enable_if< is_convertible< typename iterator_traits<_InIter >::iterator↵
_category, input_iterator_tag >::value >::type`
- `template<std::size_t __i, typename _Tuple >
using _Safe_tuple_element_t = typename enable_if<(__i< tuple_size<_Tuple >::value), tuple_element< __i,
_Tuple >>::type::type`
- `typedef atomic< bool > atomic_bool`
- `typedef atomic< char > atomic_char`
- `typedef atomic< char16_t > atomic_char16_t`
- `typedef atomic< char32_t > atomic_char32_t`
- `typedef atomic< int > atomic_int`
- `typedef atomic< int_fast16_t > atomic_int_fast16_t`
- `typedef atomic< int_fast32_t > atomic_int_fast32_t`
- `typedef atomic< int_fast64_t > atomic_int_fast64_t`
- `typedef atomic< int_fast8_t > atomic_int_fast8_t`
- `typedef atomic< int_least16_t > atomic_int_least16_t`
- `typedef atomic< int_least32_t > atomic_int_least32_t`
- `typedef atomic< int_least64_t > atomic_int_least64_t`
- `typedef atomic< int_least8_t > atomic_int_least8_t`
- `typedef atomic< intmax_t > atomic_intmax_t`

- typedef [atomic](#)< intptr_t > [atomic_intptr_t](#)
- typedef [atomic](#)< long long > [atomic_llong](#)
- typedef [atomic](#)< long > [atomic_long](#)
- typedef [atomic](#)< ptrdiff_t > [atomic_ptrdiff_t](#)
- typedef [atomic](#)< signed char > [atomic_schar](#)
- typedef [atomic](#)< short > [atomic_short](#)
- typedef [atomic](#)< size_t > [atomic_size_t](#)
- typedef [atomic](#)< unsigned char > [atomic_uchar](#)
- typedef [atomic](#)< unsigned int > [atomic_uint](#)
- typedef [atomic](#)< uint_fast16_t > [atomic_uint_fast16_t](#)
- typedef [atomic](#)< uint_fast32_t > [atomic_uint_fast32_t](#)
- typedef [atomic](#)< uint_fast64_t > [atomic_uint_fast64_t](#)
- typedef [atomic](#)< uint_fast8_t > [atomic_uint_fast8_t](#)
- typedef [atomic](#)< uint_least16_t > [atomic_uint_least16_t](#)
- typedef [atomic](#)< uint_least32_t > [atomic_uint_least32_t](#)
- typedef [atomic](#)< uint_least64_t > [atomic_uint_least64_t](#)
- typedef [atomic](#)< uint_least8_t > [atomic_uint_least8_t](#)
- typedef [atomic](#)< uintmax_t > [atomic_uintmax_t](#)
- typedef [atomic](#)< intptr_t > [atomic_intptr_t](#)
- typedef [atomic](#)< unsigned long long > [atomic_ullong](#)
- typedef [atomic](#)< unsigned long > [atomic_ulong](#)
- typedef [atomic](#)< unsigned short > [atomic_ushort](#)
- typedef [atomic](#)< wchar_t > [atomic_wchar_t](#)
- typedef [match_results](#)< const char * > **cmatch**
- typedef [regex_iterator](#)< const char * > **cregex_iterator**
- typedef [regex_token_iterator](#)< const char * > [cregex_token_iterator](#)
- typedef [sub_match](#)< const char * > [csub_match](#)
- typedef [minstd_rand0](#) **default_random_engine**
- typedef [integral_constant](#)< bool, false > [false_type](#)
- typedef [basic_filebuf](#)< char > [filebuf](#)
- typedef [basic_fstream](#)< char > [fstream](#)
- typedef [basic_ifstream](#)< char > [ifstream](#)
- template<size_t... _Idx>
using [index_sequence](#) = [integer_sequence](#)< size_t, _Idx... >
- template<typename... _Types>
using [index_sequence_for](#) = [make_index_sequence](#)< sizeof...(_Types)>
- typedef [basic_ios](#)< char > [ios](#)
- typedef [basic_iostream](#)< char > [iostream](#)
- typedef [basic_istream](#)< char > [istream](#)
- typedef [basic_istreamstream](#)< char > [istreamstream](#)
- typedef [shuffle_order_engine](#)< [minstd_rand0](#), 256 > **knuth_b**
- template<size_t _Num>
using [make_index_sequence](#) = [make_integer_sequence](#)< size_t, _Num >
- template<typename _Tp, _Tp _Num>
using [make_integer_sequence](#) = typename [_Make_integer_sequence](#)< _Tp, _Num >::__type
- typedef enum [std::memory_order](#) [memory_order](#)
- typedef [linear_congruential_engine](#)< uint_fast32_t, 48271UL, 0UL, 2147483647UL > [minstd_rand](#)
- typedef [linear_congruential_engine](#)< uint_fast32_t, 16807UL, 0UL, 2147483647UL > [minstd_rand0](#)
- typedef [mersenne_twister_engine](#)< uint_fast32_t, 32, 624, 397, 31, 0x9908b0dfUL, 11, 0xffffffffUL, 7, 0x9d2c5680UL, 15, 0xefc60000UL, 18, 1812433253UL > [mt19937](#)

- typedef [mersenne_twister_engine](#)< uint_fast64_t, 64, 312, 156, 31, 0xb5026f5aa96619e9ULL, 29, 0x5555555555555555ULL, 17, 0x71d67ffeda60000ULL, 37, 0xffff7eee00000000ULL, 43, 6364136223846793005ULL > [mt19937_64](#)
- typedef void(* [new_handler](#)) ()
- typedef [basic_ofstream](#)< char > [ofstream](#)
- typedef [basic_ostream](#)< char > [ostream](#)
- typedef [basic_ostringstream](#)< char > [ostringstream](#)
- typedef __PTRDIFF_TYPE__ [ptrdiff_t](#)
- typedef [discard_block_engine](#)< [ranlux24_base](#), 223, 23 > [ranlux24](#)
- typedef [subtract_with_carry_engine](#)< uint_fast32_t, 24, 10, 24 > [ranlux24_base](#)
- typedef [discard_block_engine](#)< [ranlux48_base](#), 389, 11 > [ranlux48](#)
- typedef [subtract_with_carry_engine](#)< uint_fast64_t, 48, 5, 12 > [ranlux48_base](#)
- template<typename _R1, typename _R2 >
using [ratio_divide](#) = typename __ratio_divide< _R1, _R2 >::type
- template<typename _R1, typename _R2 >
using [ratio_multiply](#) = typename __ratio_multiply< _R1, _R2 >::type
- typedef [basic_regex](#)< char > [regex](#)
- typedef __SIZE_TYPE__ [size_t](#)
- typedef [match_results](#)< string::const_iterator > [smatch](#)
- typedef [regex_iterator](#)< string::const_iterator > [sregex_iterator](#)
- typedef [regex_token_iterator](#)< string::const_iterator > [sregex_token_iterator](#)
- typedef [sub_match](#)< string::const_iterator > [ssub_match](#)
- typedef [basic_streambuf](#)< char > [streambuf](#)
- typedef long long [streamoff](#)
- typedef [fpos](#)< mbstate_t > [streampos](#)
- typedef [ptrdiff_t](#) [streamsize](#)
- typedef [basic_string](#)< char > [string](#)
- typedef [basic_stringbuf](#)< char > [stringbuf](#)
- typedef [basic_stringstream](#)< char > [stringstream](#)
- typedef void(* [terminate_handler](#)) ()
- typedef [integral_constant](#)< bool, true > [true_type](#)
- template<std::size_t __i, typename _Tp >
using [tuple_element_t](#) = typename [tuple_element](#)< __i, _Tp >::type
- typedef [fpos](#)< mbstate_t > [u16streampos](#)
- typedef [basic_string](#)< char16_t > [u16string](#)
- typedef [fpos](#)< mbstate_t > [u32streampos](#)
- typedef [basic_string](#)< char32_t > [u32string](#)
- typedef void(* [unexpected_handler](#)) ()
- typedef [match_results](#)< const wchar_t * > [wcmatch](#)
- typedef [regex_iterator](#)< const wchar_t * > [wcregex_iterator](#)
- typedef [regex_token_iterator](#)< const wchar_t * > [wcregex_token_iterator](#)
- typedef [sub_match](#)< const wchar_t * > [wcsub_match](#)
- typedef [basic_filebuf](#)< wchar_t > [wfilebuf](#)
- typedef [basic_fstream](#)< wchar_t > [wfstream](#)
- typedef [basic_ifstream](#)< wchar_t > [wifstream](#)
- typedef [basic_ios](#)< wchar_t > [wios](#)
- typedef [basic_iostream](#)< wchar_t > [wiostream](#)
- typedef [basic_istream](#)< wchar_t > [wistream](#)
- typedef [basic_istreamstream](#)< wchar_t > [wistreamstream](#)
- typedef [basic_ofstream](#)< wchar_t > [wofstream](#)
- typedef [basic_ostream](#)< wchar_t > [wostream](#)

- typedef [basic_ostringstream](#)< wchar_t > [wostringstream](#)
- typedef [basic_regex](#)< wchar_t > [wregex](#)
- typedef [match_results](#)< wstring::const_iterator > [wsmatch](#)
- typedef [regex_iterator](#)< wstring::const_iterator > [wsregex_iterator](#)
- typedef [regex_token_iterator](#)< wstring::const_iterator > [wsregex_token_iterator](#)
- typedef [sub_match](#)< wstring::const_iterator > [wssub_match](#)
- typedef [basic_streambuf](#)< wchar_t > [wstreambuf](#)
- typedef [fpos](#)< mbstate_t > [wstreampos](#)
- typedef [basic_string](#)< wchar_t > [wstring](#)
- typedef [basic_stringbuf](#)< wchar_t > [wstringbuf](#)
- typedef [basic_stringstream](#)< wchar_t > [wstringstream](#)

Enumerations

- enum { [_S_threshold](#) }
- enum { [_S_chunk_size](#) }
- enum { [_S_word_bit](#) }
- enum [__memory_order_modifier](#) { [__memory_order_mask](#), [__memory_order_modifier_mask](#), [__memory_order_hle_acquire](#), [__memory_order_hle_release](#) }
- enum [_ios_Fmtflags](#) { [_S_boolalpha](#), [_S_dec](#), [_S_fixed](#), [_S_hex](#), [_S_internal](#), [_S_left](#), [_S_oct](#), [_S_right](#), [_S_scientific](#), [_S_showbase](#), [_S_showpoint](#), [_S_showpos](#), [_S_skipws](#), [_S_unitbuf](#), [_S_uppercase](#), [_S_adjustfield](#), [_S_basefield](#), [_S_floatfield](#), [_S_ios_fmtflags_end](#), [_S_ios_fmtflags_max](#), [_S_ios_fmtflags_min](#) }
- enum [_ios_iostate](#) { [_S_goodbit](#), [_S_badbit](#), [_S_eofbit](#), [_S_failbit](#), [_S_ios_iostate_end](#), [_S_ios_iostate_max](#), [_S_ios_iostate_min](#) }
- enum [_ios_Openmode](#) { [_S_app](#), [_S_ate](#), [_S_bin](#), [_S_in](#), [_S_out](#), [_S_trunc](#), [_S_ios_openmode_end](#), [_S_ios_openmode_max](#), [_S_ios_openmode_min](#) }
- enum [_ios_Seekdir](#) { [_S_beg](#), [_S_cur](#), [_S_end](#), [_S_ios_seekdir_end](#) }
- enum [_Manager_operation](#) { [__get_type_info](#), [__get_functor_ptr](#), [__clone_functor](#), [__destroy_functor](#) }
- enum [_Rb_tree_color](#) { [_S_red](#), [_S_black](#) }
- enum [codecvt_mode](#) { [consume_header](#), [generate_header](#), [little_endian](#) }
- enum [cv_status](#) { [no_timeout](#), [timeout](#) }
- enum [errc](#) { [address_family_not_supported](#), [address_in_use](#), [address_not_available](#), [already_connected](#), [argument_list_too_long](#), [argument_out_of_domain](#), [bad_address](#), [bad_file_descriptor](#), [broken_pipe](#), [connection_aborted](#), [connection_already_in_progress](#), [connection_refused](#), [connection_reset](#), [cross_device_link](#), [destination_address_required](#), [device_or_resource_busy](#), [directory_not_empty](#), [executable_format_error](#), [file_exists](#), [file_too_large](#), [filename_too_long](#), [function_not_supported](#), [host_unreachable](#), [illegal_byte_sequence](#), [inappropriate_io_control_operation](#), [interrupted](#), [invalid_argument](#), [invalid_seek](#), [io_error](#), [is_a_directory](#), [message_size](#), [network_down](#), [network_reset](#), [network_unreachable](#), [no_buffer_space](#), [no_child_process](#), [no_lock_available](#), [no_message](#), [no_protocol_option](#), [no_space_on_device](#), [no_such_device_or_address](#), [no_such_device](#), [no_such_file_or_directory](#), [no_such_process](#), [not_a_directory](#), [not_a_socket](#), [not_connected](#), [not_enough_memory](#), [operation_in_progress](#), [operation_not_permitted](#), [operation_not_supported](#), [operation_would_block](#), [permission_denied](#), [protocol_not_supported](#), [read_only_file_system](#), [resource_deadlock_would_occur](#), [resource_unavailable_try_again](#), [result_out_of_range](#), [timed_out](#), [too_many_files_open_in_system](#),

- `too_many_files_open`, `too_many_links`, `too_many_symbolic_link_levels`, `wrong_protocol_type` }
- enum `float_denorm_style` { `denorm_indeterminate`, `denorm_absent`, `denorm_present` }
- enum `float_round_style` { `round_indeterminate`, `round_toward_zero`, `round_to_nearest`, `round_toward_infinity`, `round_toward_neg_infinity` }
- enum `future_errc` { `future_already_retrieved`, `promise_already_satisfied`, `no_state`, `broken_promise` }
- enum `future_status` { `ready`, `timeout`, `deferred` }
- enum `io_errc` { `stream` }
- enum `launch` { `async`, `deferred` }
- enum `memory_order` { `memory_order_relaxed`, `memory_order_consume`, `memory_order_acquire`, `memory_order_release`, `memory_order_acq_rel`, `memory_order_seq_cst` }
- enum `pointer_safety` { `relaxed`, `preferred`, `strict` }

Functions

- template<typename _CharT >
_CharT * **__add_grouping** (_CharT * __s, _CharT __sep, const char * __gbeg, size_t __gsize, const _CharT * __first, const _CharT * __last)
- template<typename _Tp >
_Tp * **__addressof** (_Tp & __r) noexcept
- template<typename _ForwardIterator, typename _BinaryPredicate >
_ForwardIterator **__adjacent_find** (_ForwardIterator __first, _ForwardIterator __last, _BinaryPredicate __↵ binary_pred)
- template<typename _RandomAccessIterator, typename _Distance, typename _Tp, typename _Compare >
void **__adjust_heap** (_RandomAccessIterator __first, _Distance __holeIndex, _Distance __len, _Tp __value, ↵ _Compare __comp)
- template<typename _InputIterator, typename _Distance >
void **__advance** (_InputIterator & __i, _Distance __n, [input_iterator_tag](#))
- template<typename _BidirectionalIterator, typename _Distance >
void **__advance** (_BidirectionalIterator & __i, _Distance __n, [bidirectional_iterator_tag](#))
- template<typename _RandomAccessIterator, typename _Distance >
void **__advance** (_RandomAccessIterator & __i, _Distance __n, [random_access_iterator_tag](#))
- template<typename _Alloc >
void **__alloc_on_copy** (_Alloc & __one, const _Alloc & __two)
- template<typename _Alloc >
_Alloc **__alloc_on_copy** (const _Alloc & __a)
- template<typename _Alloc >
void **__alloc_on_move** (_Alloc & __one, _Alloc & __two)
- template<typename _Alloc >
void **__alloc_on_swap** (_Alloc & __one, _Alloc & __two)
- template<typename _Alloc >
[__allocated_ptr](#)< _Alloc > [__allocate_guarded](#) (_Alloc & __a)
- template<typename _Tp, _Lock_policy _Lp, typename _Alloc, typename... _Args>
[__shared_ptr](#)< _Tp, _Lp > **__allocate_shared** (const _Alloc & __a, _Args &&... __args)
- __attribute** ((__always_inline__)) void atomic_thread_fence([memory_order](#) __m) noexcept
- namespace `__cxx11` **__attribute** ((__abi_tag__("cxx11")))
- template<typename _Callable, typename... _Args>
[__bind_simple_helper](#)< _Callable, _Args... >::type **__bind_simple** (_Callable && __callable, _Args &&... ↵ args)
- template<typename _Functor >
_Functor & **__callable_functor** (_Functor & __f)

- `template<typename _Member, typename _Class >`
`_Mem_fn< _Member _Class::* > __callable_func``tor (_Member _Class::*&__p)`
- `template<typename _Member, typename _Class >`
`_Mem_fn< _Member _Class::* > __callable_func``tor (_Member _Class::*const &__p)`
- `template<typename _Member, typename _Class >`
`_Mem_fn< _Member _Class::* > __callable_func``tor (_Member _Class::*volatile &__p)`
- `template<typename _Member, typename _Class >`
`_Mem_fn< _Member _Class::* > __callable_func``tor (_Member _Class::*const volatile &__p)`
- `template<typename _Facet >`
`const _Facet & __check_facet` `(const _Facet *__f)`
- `template<typename _RandomAccessIterator, typename _Distance, typename _Compare >`
`void __chunk_insertion_sort` `(_RandomAccessIterator __first, _RandomAccessIterator __last, _Distance __↵`
`chunk_size, _Compare __comp)`
- `constexpr memory_order __cmpexch_failure_order` `(memory_order __m) noexcept`
- `constexpr memory_order __cmpexch_failure_order2` `(memory_order __m) noexcept`
- `template<typename _Tp >`
`_Tp __complex_abs` `(const complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > __complex_acos` `(const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > __complex_acosh` `(const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`_Tp __complex_arg` `(const complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > __complex_asin` `(const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > __complex_asinh` `(const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > __complex_atan` `(const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > __complex_atanh` `(const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > __complex_cos` `(const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > __complex_cosh` `(const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > __complex_exp` `(const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > __complex_log` `(const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > __complex_pow` `(const complex< _Tp > &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`complex< _Tp > __complex_pow_unsigned` `(complex< _Tp > __x, unsigned __n)`
- `template<typename _Tp >`
`std::complex< _Tp > __complex_proj` `(const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > __complex_sin` `(const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > __complex_sinh` `(const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > __complex_sqrt` `(const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > __complex_tan` `(const complex< _Tp > &__z)`

- `template<typename _Tp >`
`complex< _Tp > __complex_tanh (const complex< _Tp > &__z)`
- `int __convert_from_v (const __c_locale &__cloc __attribute__((__unused__)), char *__out, const int __size __attribute__((__unused__)), const char *__fmt,...)`
- `template<typename _Tp >`
`void __convert_to_v (const char *, _Tp &, ios_base::iostate &, const __c_locale &) throw ()`
- `template<>`
`void __convert_to_v (const char *, float &, ios_base::iostate &, const __c_locale &) throw ()`
- `template<>`
`void __convert_to_v (const char *, double &, ios_base::iostate &, const __c_locale &) throw ()`
- `template<>`
`void __convert_to_v (const char *, long double &, ios_base::iostate &, const __c_locale &) throw ()`
- `template<bool _IsMove, typename _II, typename _OI >`
`_OI __copy_move_a (_II __first, _II __last, _OI __result)`
- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, ostreambuf_iterator< _CharT > >::__type __copy_move_a2 (_CharT * __first, _CharT * __last, ostreambuf_iterator< _CharT > __result)`
- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, ostreambuf_iterator< _CharT > >::__type __copy_move_a2 (const _CharT * __first, const _CharT * __last, ostreambuf_iterator< _CharT > __result)`
- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, _CharT * >::__type __copy_move_a2 (istreambuf_iterator< _CharT > __first, istreambuf_iterator< _CharT > __last, _CharT * __result)`
- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, ostreambuf_iterator< _CharT, char_traits< _CharT > > >::__type __copy_move_a2 (_CharT *, _CharT *, ostreambuf_iterator< _CharT, char_traits< _CharT > > >)`
- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, ostreambuf_iterator< _CharT, char_traits< _CharT > > >::__type __copy_move_a2 (const _CharT *, const _CharT *, ostreambuf_iterator< _CharT, char_traits< _CharT > > >)`
- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, _CharT * >::__type __copy_move_a2 (istreambuf_iterator< _CharT, char_traits< _CharT > >, istreambuf_iterator< _CharT, char_traits< _CharT > >, _CharT *)`
- `template<bool _IsMove, typename _II, typename _OI >`
`_OI __copy_move_a2 (_II __first, _II __last, _OI __result)`
- `template<bool _IsMove, typename _BI1, typename _BI2 >`
`_BI2 __copy_move_backward_a (_BI1 __first, _BI1 __last, _BI2 __result)`
- `template<bool _IsMove, typename _BI1, typename _BI2 >`
`_BI2 __copy_move_backward_a2 (_BI1 __first, _BI1 __last, _BI2 __result)`
- `template<typename _InputIterator, typename _Size, typename _OutputIterator >`
`_OutputIterator __copy_n (_InputIterator __first, _Size __n, _OutputIterator __result, input_iterator_tag)`
- `template<typename _RandomAccessIterator, typename _Size, typename _OutputIterator >`
`_OutputIterator __copy_n (_RandomAccessIterator __first, _Size __n, _OutputIterator __result, random_access_iterator_tag)`
- `template<typename _CharT, typename _Traits >`
`streamsize __copy_streambufs (basic_streambuf< _CharT, _Traits > * __sbin, basic_streambuf< _CharT, _Traits > * __sbout)`
- `template<typename _CharT, typename _Traits >`
`streamsize __copy_streambufs_eof (basic_streambuf< _CharT, _Traits > *, basic_streambuf< _CharT, _Traits > *, bool &)`

- `template<>`
`streamsize __copy_streambufs_eof (basic_streambuf< char > *__sbin, basic_streambuf< char > *__sbout,`
`bool &__ineof)`
- `template<>`
`streamsize __copy_streambufs_eof (basic_streambuf< wchar_t > *__sbin, basic_streambuf< wchar_t > *__`
`__sbout, bool &__ineof)`
- `template<typename _InputIterator, typename _Predicate >`
`iterator_traits< _InputIterator >::difference_type __count_if (_InputIterator __first, _InputIterator __last, _`
`Predicate __pred)`
- `template<typename _Signature, typename _Fn, typename _Alloc >`
`static shared_ptr< __future_base::Task_state_base< _Signature > > __create_task_state (_Fn &&__fn,`
`const _Alloc &__a)`
- `constexpr size_t __deque_buf_size (size_t __size)`
- `template<typename _InputIterator >`
`iterator_traits< _InputIterator >::difference_type __distance (_InputIterator __first, _InputIterator __last, input_`
`__iterator_tag)`
- `template<typename _RandomAccessIterator >`
`iterator_traits< _RandomAccessIterator >::difference_type __distance (_RandomAccessIterator __first, _`
`RandomAccessIterator __last, random_access_iterator_tag)`
- `template<typename _Alloc >`
`void __do_alloc_on_copy (_Alloc &__one, const _Alloc &__two, true_type)`
- `template<typename _Alloc >`
`void __do_alloc_on_copy (_Alloc &, const _Alloc &, false_type)`
- `template<typename _Alloc >`
`void __do_alloc_on_move (_Alloc &__one, _Alloc &__two, true_type)`
- `template<typename _Alloc >`
`void __do_alloc_on_move (_Alloc &, _Alloc &, false_type)`
- `template<typename _Alloc >`
`void __do_alloc_on_swap (_Alloc &__one, _Alloc &__two, true_type)`
- `template<typename _Alloc >`
`void __do_alloc_on_swap (_Alloc &, _Alloc &, false_type)`
- `template<typename _OutStr, typename _InChar, typename _Codecvt, typename _State, typename _Fn >`
`bool __do_str_codecvt (const _InChar *__first, const _InChar *__last, _OutStr &__outstr, const _Codecvt &__`
`__cvt, _State &__state, size_t &__count, _Fn __fn)`
- `template<typename _Tp1, typename _Tp2 >`
`void __enable_shared_from_this_helper (const __shared_count<> &__pn, const enable_shared_from_this<`
`__Tp1 > *__pe, const _Tp2 *__px) noexcept`
- `template<_Lock_policy _Lp, typename _Tp1, typename _Tp2 >`
`void __enable_shared_from_this_helper (const __shared_count< _Lp > &, const __enable_shared_from_`
`this< _Tp1, _Lp > *, const _Tp2 *) noexcept`
- `template<_Lock_policy _Lp>`
`void __enable_shared_from_this_helper (const __shared_count< _Lp > &,...) noexcept`
- `template<_Lock_policy _Lp1, typename _Tp1, typename _Tp2 >`
`void __enable_shared_from_this_helper (const __shared_count< _Lp1 > &__pn, const __enable_shared_`
`from_this< _Tp1, _Lp1 > *__pe, const _Tp2 *__px) noexcept`
- `template<typename _II1, typename _II2 >`
`bool __equal_aux (_II1 __first1, _II1 __last1, _II2 __first2)`
- `template<typename _ForwardIterator, typename _Tp, typename _CompareItTp, typename _CompareTpIt >`
`pair< _ForwardIterator, _ForwardIterator > __equal_range (_ForwardIterator __first, _ForwardIterator __last,`
`const _Tp &__val, _CompareItTp __comp_it_val, _CompareTpIt __comp_val_it)`
- `template<typename _Tp, typename _Up = _Tp>`
`_Tp __exchange (_Tp &__obj, _Up &&__new_val)`

- `template<typename _ForwardIterator, typename _Tp >`
`__gnu_cxx::__enable_if<!_is_scalar< _Tp >::__value, void >::__type __fill_a (_ForwardIterator __first, _↔`
`ForwardIterator __last, const _Tp &__value)`
- `template<typename _ForwardIterator, typename _Tp >`
`__gnu_cxx::__enable_if< _is_scalar< _Tp >::__value, void >::__type __fill_a (_ForwardIterator __first, _↔`
`ForwardIterator __last, const _Tp &__value)`
- `template<typename _Tp >`
`__gnu_cxx::__enable_if< _is_byte< _Tp >::__value, void >::__type __fill_a (_Tp * __first, _Tp * __last, const`
`_Tp &__c)`
- `void __fill_bvector (_Bit_iterator __first, _Bit_iterator __last, bool __x)`
- `template<typename _OutputIterator, typename _Size, typename _Tp >`
`__gnu_cxx::__enable_if<!_is_scalar< _Tp >::__value, _OutputIterator >::__type __fill_n_a (_OutputIterator`
`__first, _Size __n, const _Tp &__value)`
- `template<typename _OutputIterator, typename _Size, typename _Tp >`
`__gnu_cxx::__enable_if< _is_scalar< _Tp >::__value, _OutputIterator >::__type __fill_n_a (_OutputIterator`
`__first, _Size __n, const _Tp &__value)`
- `template<typename _Size, typename _Tp >`
`__gnu_cxx::__enable_if< _is_byte< _Tp >::__value, _Tp * >::__type __fill_n_a (_Tp * __first, _Size __n,`
`const _Tp &__c)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void __final_insertion_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`_ForwardIterator1 __find_end (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2,`
`_ForwardIterator2 __last2, forward_iterator_tag, forward_iterator_tag, _BinaryPredicate __comp)`
- `template<typename _BidirectionalIterator1, typename _BidirectionalIterator2, typename _BinaryPredicate >`
`_BidirectionalIterator1 __find_end (_BidirectionalIterator1 __first1, _BidirectionalIterator1 __last1, _↔`
`BidirectionalIterator2 __first2, _BidirectionalIterator2 __last2, bidirectional_iterator_tag, bidirectional_iterator_tag,`
`_BinaryPredicate __comp)`
- `template<typename _InputIterator, typename _Predicate >`
`_InputIterator __find_if (_InputIterator __first, _InputIterator __last, _Predicate __pred, input_iterator_tag)`
- `template<typename _RandomAccessIterator, typename _Predicate >`
`_RandomAccessIterator __find_if (_RandomAccessIterator __first, _RandomAccessIterator __last, _Predicate _↔`
`__pred, random_access_iterator_tag)`
- `template<typename _Iterator, typename _Predicate >`
`_Iterator __find_if (_Iterator __first, _Iterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _Predicate >`
`_InputIterator __find_if_not (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _Predicate, typename _Distance >`
`_InputIterator __find_if_not_n (_InputIterator __first, _Distance &__len, _Predicate __pred)`
- `template<typename _EuclideanRingElement >`
`_EuclideanRingElement __gcd (_EuclideanRingElement __m, _EuclideanRingElement __n)`
- `template<std::size_t __i, typename _Head, typename... _Tail>`
`constexpr _Head & __get_helper (_Tuple_impl< __i, _Head, _Tail... > &__t) noexcept`
- `template<std::size_t __i, typename _Head, typename... _Tail>`
`constexpr const _Head & __get_helper (const _Tuple_impl< __i, _Head, _Tail... > &__t) noexcept`
- `template<typename _Head, size_t __i, typename... _Tail>`
`constexpr _Head & __get_helper2 (_Tuple_impl< __i, _Head, _Tail... > &__t) noexcept`
- `template<typename _Head, size_t __i, typename... _Tail>`
`constexpr const _Head & __get_helper2 (const _Tuple_impl< __i, _Head, _Tail... > &__t) noexcept`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void __heap_select (_RandomAccessIterator __first, _RandomAccessIterator __middle, _RandomAccessIterator`
`__last, _Compare __comp)`

- `template<typename _Tp >`
`size_t __iconv_adapter (size_t(*__func)(iconv_t, _Tp, size_t *, char **, size_t *), iconv_t __cd, char **__inbuf,`
`size_t *__inbytes, char **__outbuf, size_t *__outbytes)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Compare >`
`bool __includes (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2,`
`_Compare __comp)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`void __inplace_merge (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last,`
`_Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void __inplace_stable_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void __insertion_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _CharT, typename _ValueT >`
`int __int_to_char (_CharT *__bufend, _ValueT __v, const _CharT *__lit, ios_base::fmtflags __flags, bool __dec)`
- `template<typename _RandomAccessIterator, typename _Size, typename _Compare >`
`void __introselct (_RandomAccessIterator __first, _RandomAccessIterator __nth, _RandomAccessIterator __↵`
`__last, _Size __depth_limit, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Size, typename _Compare >`
`void __introsort_loop (_RandomAccessIterator __first, _RandomAccessIterator __last, _Size __depth_limit, __↵`
`_Compare __comp)`
- `template<typename _Tp, typename _Up = typename __inv_unwrap<_Tp>::type>`
`_Up && __invfwd (typename remove_reference<_Tp>::type &__t) noexcept`
- `template<typename _Callable, typename... _Args>`
`result_of<_Callable &&(_Args &&...)>::type __invoke (_Callable &&__fn, _Args &&... __args)`
- `template<typename _Res, typename _Fn, typename... _Args>`
`_Res __invoke_impl (_invoke_other, _Fn &&__f, _Args &&... __args) noexcept(noexcept(std::forward<_Fn`
`>(__f)(std::forward<_Args>(__args)...))`
- `template<typename _Res, typename _MemFun, typename _Tp, typename... _Args>`
`_Res __invoke_impl (__invoke_memfun_ref, _MemFun &&__f, _Tp &&__t, _Args &&... __args) noexcept(noexcept((__↵`
`__invfwd<_Tp>(__t).*_f)(std::forward<_Args>(__args)...))`
- `template<typename _Res, typename _MemFun, typename _Tp, typename... _Args>`
`_Res __invoke_impl (__invoke_memfun_deref, _MemFun &&__f, _Tp &&__t, _Args &&... __args) noexcept(noexcept(((std::↵`
`::forward<_Tp>(__t)).*_f)(std::forward<_Args>(__args)...))`
- `template<typename _Res, typename _MemPtr, typename _Tp >`
`_Res __invoke_impl (__invoke_memobj_ref, _MemPtr &&__f, _Tp &&__t) noexcept(noexcept(__invfwd<_Tp`
`>(__t).*_f))`
- `template<typename _Res, typename _MemPtr, typename _Tp >`
`_Res __invoke_impl (__invoke_memobj_deref, _MemPtr &&__f, _Tp &&__t) noexcept(noexcept(((std::forward<`
`_Tp>(__t)).*_f))`
- `template<typename _RandomAccessIterator, typename _Distance >`
`bool __is_heap (_RandomAccessIterator __first, _Distance __n)`
- `template<typename _RandomAccessIterator, typename _Compare, typename _Distance >`
`bool __is_heap (_RandomAccessIterator __first, _Compare __comp, _Distance __n)`
- `template<typename _RandomAccessIterator >`
`bool __is_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`bool __is_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Distance, typename _Compare >`
`_Distance __is_heap_until (_RandomAccessIterator __first, _Distance __n, _Compare __comp)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`bool __is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, __↵`
`_BinaryPredicate __pred)`

- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`bool __is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, ↵`
`_ForwardIterator2 __last2, _BinaryPredicate __pred)`
- `template<typename _ForwardIterator, typename _Compare >`
`_ForwardIterator __is_sorted_until (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _Iter >`
`iterator_traits< _Iter >::iterator_category __iterator_category (const _Iter &)`
- `template<typename _I1, typename _I2 >`
`bool __lexicographical_compare_aux (_I1 __first1, _I1 __last1, _I2 __first2, _I2 __last2)`
- `template<typename _I1, typename _I2, typename _Compare >`
`bool __lexicographical_compare_impl (_I1 __first1, _I1 __last1, _I2 __first2, _I2 __last2, _Compare __↵`
`comp)`
- `constexpr int __lg (int __n)`
- `constexpr unsigned __lg (unsigned __n)`
- `constexpr long __lg (long __n)`
- `constexpr unsigned long __lg (unsigned long __n)`
- `constexpr long long __lg (long long __n)`
- `constexpr unsigned long long __lg (unsigned long long __n)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`_ForwardIterator __lower_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare`
`__comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void __make_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _Iterator, typename _ReturnType = typename conditional<__move_if_noexcept_cond <typename iterator_traits<↵`
`Iterator>::value_type>::value, _Iterator, move_iterator<_Iterator>>::type>`
`_ReturnType __make_move_if_noexcept_iterator (_Iterator __i)`
- `template<typename _Tp, typename _ReturnType = typename conditional<__move_if_noexcept_cond<_Tp>::value, const _Tp*, move↵`
`_Iterator<_Tp*>>::type>`
`_ReturnType __make_move_if_noexcept_iterator (_Tp *__i)`
- `template<typename _Iterator >`
`reverse_iterator< _Iterator > __make_reverse_iterator (_Iterator __i)`
- `template<typename _Tp, _Lock_policy _Lp, typename... _Args>`
`__shared_ptr< _Tp, _Lp > __make_shared (_Args &&...__args)`
- `template<typename _ForwardIterator, typename _Compare >`
`_GLIBCXX14_CONSTEXPR _ForwardIterator __max_element (_ForwardIterator __first, _ForwardIterator __↵`
`last, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator __merge (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2`
`__last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _BidirectionalIterator, typename _Distance, typename _Pointer, typename _Compare >`
`void __merge_adaptive (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __↵`
`last, _Distance __len1, _Distance __len2, _Pointer __buffer, _Distance __buffer_size, _Compare __comp)`
- `template<typename _RandomAccessIterator1, typename _RandomAccessIterator2, typename _Distance, typename _Compare >`
`void __merge_sort_loop (_RandomAccessIterator1 __first, _RandomAccessIterator1 __last, _Random↵`
`AccessIterator2 __result, _Distance __step_size, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Pointer, typename _Compare >`
`void __merge_sort_with_buffer (_RandomAccessIterator __first, _RandomAccessIterator __last, _Pointer __↵`
`buffer, _Compare __comp)`
- `template<typename _BidirectionalIterator, typename _Distance, typename _Compare >`
`void __merge_without_buffer (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator`
`__last, _Distance __len1, _Distance __len2, _Compare __comp)`

- `template<typename _ForwardIterator, typename _Compare >`
`_GLIBCXX14_CONSTEXPR _ForwardIterator __min_element (_ForwardIterator __first, _ForwardIterator __last,`
`_Compare __comp)`
- `template<typename _ForwardIterator, typename _Compare >`
`_GLIBCXX14_CONSTEXPR pair< _ForwardIterator, _ForwardIterator > __minmax_element (_ForwardIterator`
`__first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`pair< _InputIterator1, _InputIterator2 > __mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2`
`__first2, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`pair< _InputIterator1, _InputIterator2 > __mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2`
`__first2, _InputIterator2 __last2, _BinaryPredicate __binary_pred)`
- `template<typename _Iterator >`
`_Iterator __miter_base (_Iterator __it)`
- `template<typename _Iterator >`
`auto __miter_base (reverse_iterator< _Iterator > __it) -> decltype(__make_reverse_iterator(__miter_base(__it.base())))`
- `template<typename _Iterator >`
`auto __miter_base (move_iterator< _Iterator > __it) -> decltype(__miter_base(__it.base()))`
- `template<typename _Iterator, typename _Compare >`
`void __move_median_to_first (_Iterator __result, _Iterator __a, _Iterator __b, _Iterator __c, _Compare __comp)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Compare >`
`_OutputIterator __move_merge (_InputIterator __first1, _InputIterator __last1, _InputIterator __first2, _InputIterator`
`__last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`void __move_merge_adaptive (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2`
`__last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _BidirectionalIterator1, typename _BidirectionalIterator2, typename _BidirectionalIterator3, typename _Compare >`
`void __move_merge_adaptive_backward (_BidirectionalIterator1 __first1, _BidirectionalIterator1 __last1, _BidirectionalIterator2`
`__first2, _BidirectionalIterator2 __last2, _BidirectionalIterator3 __result, _Compare __comp)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`bool __next_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare __comp)`
- `template<typename _Iterator >`
`_Iterator __niter_base (_Iterator __it)`
- `template<typename _Iterator >`
`auto __niter_base (reverse_iterator< _Iterator > __it) -> decltype(__make_reverse_iterator(__niter_base(__it.base())))`
- `template<typename _Iterator, typename _Container >`
`_Iterator __niter_base (__gnu_cxx::__normal_iterator< _Iterator, _Container > __it)`
- `template<typename _Iterator >`
`auto __niter_base (move_iterator< _Iterator > __it) -> decltype(make_move_iterator(__niter_base(__it.base())))`
- `template<typename _CharT, typename _Traits >`
`void __ostream_fill (basic_ostream< _CharT, _Traits > &__out, streamsize __n)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & __ostream_insert (basic_ostream< _CharT, _Traits > &__out, const _CharT * __s,`
`streamsize __n)`
- `template<typename _CharT, typename _Traits >`
`void __ostream_write (basic_ostream< _CharT, _Traits > &__out, const _CharT * __s, streamsize __n)`
- `template<typename _Alloc >`
`__outermost_type< _Alloc >::type & __outermost (_Alloc & __a)`

- `template<typename _RandomAccessIterator, typename _Compare >`
`void __partial_sort (_RandomAccessIterator __first, _RandomAccessIterator __middle, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _Compare >`
`_RandomAccessIterator __partial_sort_copy (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __result_first, _RandomAccessIterator __result_last, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator __partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred, forward_iterator_tag)`
- `template<typename _BidirectionalIterator, typename _Predicate >`
`_BidirectionalIterator __partition (_BidirectionalIterator __first, _BidirectionalIterator __last, _Predicate __pred, bidirectional_iterator_tag)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void __pop_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _RandomAccessIterator __result, _Compare __comp)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`bool __prev_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Distance, typename _Tp, typename _Compare >`
`void __push_heap (_RandomAccessIterator __first, _Distance __holeIndex, _Distance __topIndex, _Tp __value, _Compare __comp)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate >`
`_OutputIterator __remove_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Predicate __pred)`
- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator __remove_if (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate, typename _Tp >`
`_OutputIterator __replace_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Predicate __pred, const _Tp &__new_value)`
- `template<typename _Ex >`
`__rethrow_if_nested_cond< _Ex > __rethrow_if_nested_impl (const _Ex *__ptr)`
- `void __rethrow_if_nested_impl (const void *)`
- `template<typename _BidirectionalIterator >`
`void __reverse (_BidirectionalIterator __first, _BidirectionalIterator __last, bidirectional_iterator_tag)`
- `template<typename _RandomAccessIterator >`
`void __reverse (_RandomAccessIterator __first, _RandomAccessIterator __last, random_access_iterator_tag)`
- `template<typename _BidirectionalIterator1, typename _BidirectionalIterator2, typename _Distance >`
`_BidirectionalIterator1 __rotate_adaptive (_BidirectionalIterator1 __first, _BidirectionalIterator1 __middle, _BidirectionalIterator1 __last, _Distance __len1, _Distance __len2, _BidirectionalIterator2 __buffer, _Distance __buffer_size)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`_ForwardIterator1 __search (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2 __last2, _BinaryPredicate __predicate)`
- `template<typename _ForwardIterator, typename _Integer, typename _UnaryPredicate >`
`_ForwardIterator __search_n (_ForwardIterator __first, _ForwardIterator __last, _Integer __count, _UnaryPredicate __unary_pred)`
- `template<typename _ForwardIterator, typename _Integer, typename _UnaryPredicate >`
`_ForwardIterator __search_n_aux (_ForwardIterator __first, _ForwardIterator __last, _Integer __count, _UnaryPredicate __unary_pred, std::forward_iterator_tag)`
- `template<typename _RandomAccessIterator, typename _Integer, typename _UnaryPredicate >`
`_RandomAccessIterator __search_n_aux (_RandomAccessIterator __first, _RandomAccessIterator __last, _Integer __count, _UnaryPredicate __unary_pred, std::random_access_iterator_tag)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator __set_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`

- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator set_intersection (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, ↵`
`_InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator set_symmetric_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2`
`__first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator set_union (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, ↵`
`_InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void sort_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator stable_partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _ForwardIterator, typename _Pointer, typename _Predicate, typename _Distance >`
`_ForwardIterator stable_partition_adaptive (_ForwardIterator __first, _ForwardIterator __last, _Predicate ↵`
`__pred, _Distance __len, _Pointer __buffer, _Distance __buffer_size)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void stable_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Pointer, typename _Distance, typename _Compare >`
`void stable_sort_adaptive (_RandomAccessIterator __first, _RandomAccessIterator __last, _Pointer __buffer,`
`_Distance __buffer_size, _Compare __comp)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool str_codecvt_in (const char * __first, const char * __last, basic_string< _CharT, _Traits, _Alloc > &↵`
`__outstr, const codecvt< _CharT, char, _State > & __cvt, _State & __state, size_t & __count)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool str_codecvt_in (const char * __first, const char * __last, basic_string< _CharT, _Traits, _Alloc > &↵`
`__outstr, const codecvt< _CharT, char, _State > & __cvt)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool str_codecvt_out (const _CharT * __first, const _CharT * __last, basic_string< char, _Traits, _Alloc >`
`& __outstr, const codecvt< _CharT, char, _State > & __cvt, _State & __state, size_t & __count)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool str_codecvt_out (const _CharT * __first, const _CharT * __last, basic_string< char, _Traits, _Alloc >`
`& __outstr, const codecvt< _CharT, char, _State > & __cvt)`
- `void throw_bad_alloc (void) __attribute__((__noreturn__))`
- `void throw_bad_cast (void) __attribute__((__noreturn__))`
- `void throw_bad_exception (void) __attribute__((__noreturn__))`
- `void throw_bad_function_call () __attribute__((__noreturn__))`
- `void throw_bad_typeid (void) __attribute__((__noreturn__))`
- `void throw_bad_weak_ptr ()`
- `void throw_domain_error (const char *) __attribute__((__noreturn__))`
- `void throw_future_error (int) __attribute__((__noreturn__))`
- `void throw_invalid_argument (const char *) __attribute__((__noreturn__))`
- `void throw_ios_failure (const char *) __attribute__((__noreturn__))`
- `void throw_length_error (const char *) __attribute__((__noreturn__))`
- `void throw_logic_error (const char *) __attribute__((__noreturn__))`
- `void throw_out_of_range (const char *) __attribute__((__noreturn__))`
- `void throw_out_of_range_fmt (const char *,...) __attribute__((__noreturn__)) __attribute__((__format__(↵`
`__gnu_printf__`
- `void throw_overflow_error (const char *) __attribute__((__noreturn__))`
- `void throw_range_error (const char *) __attribute__((__noreturn__))`

- void `__throw_regex_error` ([regex_constants::error_type](#) __ecode)
- void `__throw_regex_error` ([regex_constants::error_type](#) __ecode, const char * __what)
- void `__throw_runtime_error` (const char *) __attribute__((__noreturn__))
- void `__throw_system_error` (int) __attribute__((__noreturn__))
- void `__throw_underflow_error` (const char *) __attribute__((__noreturn__))
- template<typename _Tp >
void `__throw_with_nested_impl` (_Tp && __t, [true_type](#))
- template<typename _Tp >
void `__throw_with_nested_impl` (_Tp && __t, [false_type](#))
- template<typename _RandomAccessIterator, typename _Compare >
void `__unguarded_insertion_sort` (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare <= __comp)
- template<typename _RandomAccessIterator, typename _Compare >
void `__unguarded_linear_insert` (_RandomAccessIterator __last, _Compare __comp)
- template<typename _RandomAccessIterator, typename _Compare >
_RandomAccessIterator `__unguarded_partition` (_RandomAccessIterator __first, _RandomAccessIterator __last, _RandomAccessIterator __pivot, _Compare __comp)
- template<typename _RandomAccessIterator, typename _Compare >
_RandomAccessIterator `__unguarded_partition_pivot` (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)
- template<typename _Pointer, typename _ForwardIterator >
void `__uninitialized_construct_buf` (_Pointer __first, _Pointer __last, _ForwardIterator __seed)
- template<typename _InputIterator, typename _ForwardIterator, typename _Allocator >
_ForwardIterator `__uninitialized_copy_a` (_InputIterator __first, _InputIterator __last, _ForwardIterator __result, _Allocator & __alloc)
- template<typename _InputIterator, typename _ForwardIterator, typename _Tp >
_ForwardIterator `__uninitialized_copy_a` (_InputIterator __first, _InputIterator __last, _ForwardIterator __result, [allocator](#)< _Tp > &)
- template<typename _InputIterator1, typename _InputIterator2, typename _ForwardIterator, typename _Allocator >
_ForwardIterator `__uninitialized_copy_move` (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _ForwardIterator __result, _Allocator & __alloc)
- template<typename _InputIterator, typename _Size, typename _ForwardIterator >
_ForwardIterator `__uninitialized_copy_n` (_InputIterator __first, _Size __n, _ForwardIterator __result, [input_iterator_tag](#))
- template<typename _RandomAccessIterator, typename _Size, typename _ForwardIterator >
_ForwardIterator `__uninitialized_copy_n` (_RandomAccessIterator __first, _Size __n, _ForwardIterator __result, [random_access_iterator_tag](#))
- template<typename _ForwardIterator >
void `__uninitialized_default` (_ForwardIterator __first, _ForwardIterator __last)
- template<typename _ForwardIterator, typename _Allocator >
void `__uninitialized_default_a` (_ForwardIterator __first, _ForwardIterator __last, _Allocator & __alloc)
- template<typename _ForwardIterator, typename _Tp >
void `__uninitialized_default_a` (_ForwardIterator __first, _ForwardIterator __last, [allocator](#)< _Tp > &)
- template<typename _ForwardIterator, typename _Size >
_ForwardIterator `__uninitialized_default_n` (_ForwardIterator __first, _Size __n)
- template<typename _ForwardIterator, typename _Size, typename _Allocator >
_ForwardIterator `__uninitialized_default_n_a` (_ForwardIterator __first, _Size __n, _Allocator & __alloc)
- template<typename _ForwardIterator, typename _Size, typename _Tp >
_ForwardIterator `__uninitialized_default_n_a` (_ForwardIterator __first, _Size __n, [allocator](#)< _Tp > &)
- template<typename _ForwardIterator, typename _Tp, typename _Allocator >
void `__uninitialized_fill_a` (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __x, _Allocator & __alloc)

- `template<typename _ForwardIterator, typename _Tp, typename _Tp2 >`
`void __uninitialized_fill_a (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__x, allocator< _Tp2 > &)`
- `template<typename _ForwardIterator, typename _Tp, typename _InputIterator, typename _Allocator >`
`_ForwardIterator __uninitialized_fill_move (_ForwardIterator __result, _ForwardIterator __mid, const _Tp &__x, _InputIterator __first, _InputIterator __last, _Allocator &__alloc)`
- `template<typename _ForwardIterator, typename _Size, typename _Tp, typename _Allocator >`
`_ForwardIterator __uninitialized_fill_n_a (_ForwardIterator __first, _Size __n, const _Tp &__x, _Allocator &__alloc)`
- `template<typename _ForwardIterator, typename _Size, typename _Tp, typename _Tp2 >`
`_ForwardIterator __uninitialized_fill_n_a (_ForwardIterator __first, _Size __n, const _Tp &__x, allocator< _Tp2 > &)`
- `template<typename _InputIterator, typename _ForwardIterator, typename _Allocator >`
`_ForwardIterator __uninitialized_move_a (_InputIterator __first, _InputIterator __last, _ForwardIterator __result, _Allocator &__alloc)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _ForwardIterator, typename _Allocator >`
`_ForwardIterator __uninitialized_move_copy (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _ForwardIterator __result, _Allocator &__alloc)`
- `template<typename _InputIterator, typename _ForwardIterator, typename _Tp, typename _Allocator >`
`void __uninitialized_move_fill (_InputIterator __first1, _InputIterator __last1, _ForwardIterator __first2, _ForwardIterator __last2, const _Tp &__x, _Allocator &__alloc)`
- `template<typename _InputIterator, typename _ForwardIterator, typename _Allocator >`
`_ForwardIterator __uninitialized_move_if_noexcept_a (_InputIterator __first, _InputIterator __last, _ForwardIterator __result, _Allocator &__alloc)`
- `template<typename _ForwardIterator, typename _BinaryPredicate >`
`_ForwardIterator __unique (_ForwardIterator __first, _ForwardIterator __last, BinaryPredicate __binary_pred)`
- `template<typename _ForwardIterator, typename _OutputIterator, typename _BinaryPredicate >`
`_OutputIterator __unique_copy (_ForwardIterator __first, _ForwardIterator __last, _OutputIterator __result, BinaryPredicate __binary_pred, forward_iterator_tag, output_iterator_tag)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryPredicate >`
`_OutputIterator __unique_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result, BinaryPredicate __binary_pred, input_iterator_tag, output_iterator_tag)`
- `template<typename _InputIterator, typename _ForwardIterator, typename _BinaryPredicate >`
`_ForwardIterator __unique_copy (_InputIterator __first, _InputIterator __last, _ForwardIterator __result, BinaryPredicate __binary_pred, input_iterator_tag, forward_iterator_tag)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`_ForwardIterator __upper_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, Compare __comp)`
- `template<typename _Tp, typename _Alloc, typename... _Args>`
`__uses_alloc_t< _Tp, _Alloc, _Args... > __use_alloc (const _Alloc &__a)`
- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __a, _Array< bool > __m, size_t __n, _Array< _Tp > __b, _Array< bool > __k)`
- `template<typename _Tp, class _Dom >`
`void __valarray_copy (const Expr< _Dom, _Tp > &__e, size_t __n, _Array< _Tp > __a)`
- `template<typename _Tp, class _Dom >`
`void __valarray_copy (const Expr< _Dom, _Tp > &__e, size_t __n, _Array< _Tp > __a, size_t __s)`
- `template<typename _Tp, class _Dom >`
`void __valarray_copy (const Expr< _Dom, _Tp > &__e, size_t __n, _Array< _Tp > __a, _Array< size_t > __i)`

- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __e, _Array< size_t > __f, size_t __n, _Array< _Tp > __a, _Array< size_t > __i)`
- `template<typename _Tp, class _Dom >`
`void __valarray_copy (const _Expr< _Dom, _Tp > &__e, size_t __n, _Array< _Tp > __a, _Array< bool > __m)`
- `template<typename _Tp >`
`void __valarray_copy (const _Tp *__restrict __a, size_t __n, _Tp *__restrict __b)`
- `template<typename _Tp >`
`void __valarray_copy (const _Tp *__restrict __a, size_t __n, size_t __s, _Tp *__restrict __b)`
- `template<typename _Tp >`
`void __valarray_copy (const _Tp *__restrict __a, _Tp *__restrict __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void __valarray_copy (const _Tp *__restrict __src, size_t __n, size_t __s1, _Tp *__restrict __dst, size_t __s2)`
- `template<typename _Tp >`
`void __valarray_copy (const _Tp *__restrict __a, const size_t *__restrict __i, _Tp *__restrict __b, size_t __n)`
- `template<typename _Tp >`
`void __valarray_copy (const _Tp *__restrict __a, size_t __n, _Tp *__restrict __b, const size_t *__restrict __i)`
- `template<typename _Tp >`
`void __valarray_copy (const _Tp *__restrict __src, size_t __n, const size_t *__restrict __i, _Tp *__restrict __dst, const size_t *__restrict __j)`
- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __a, size_t __n, size_t __s1, _Array< _Tp > __b, size_t __s2)`
- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void __valarray_copy (_Array< _Tp > __src, size_t __n, _Array< size_t > __i, _Array< _Tp > __dst, _Array< size_t > __j)`
- `template<typename _Tp >`
`void __valarray_copy_construct (const _Tp *__b, const _Tp *__e, _Tp *__restrict __o)`
- `template<typename _Tp >`
`void __valarray_copy_construct (const _Tp *__restrict __a, size_t __n, size_t __s, _Tp *__restrict __o)`
- `template<typename _Tp >`
`void __valarray_copy_construct (const _Tp *__restrict __a, const size_t *__restrict __i, _Tp *__restrict __o, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void __valarray_copy_construct (const _Expr< _Dom, _Tp > &__e, size_t __n, _Array< _Tp > __a)`
- `template<typename _Tp >`
`void __valarray_copy_construct (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void __valarray_copy_construct (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void __valarray_copy_construct (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`

- `template<typename _Tp >`
`void __valarray_default_construct (_Tp *__b, _Tp *__e)`
- `template<typename _Tp >`
`void __valarray_destroy_elements (_Tp *__b, _Tp *__e)`
- `template<typename _Tp >`
`void __valarray_fill (_Array< _Tp > __a, size_t __n, _Array< bool > __m, const _Tp &__t)`
- `template<typename _Tp >`
`void __valarray_fill (_Tp *__restrict __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void __valarray_fill (_Tp *__restrict __a, size_t __n, size_t __s, const _Tp &__t)`
- `template<typename _Tp >`
`void __valarray_fill (_Tp *__restrict __a, const size_t *__restrict __i, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void __valarray_fill (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void __valarray_fill (_Array< _Tp > __a, size_t __n, size_t __s, const _Tp &__t)`
- `template<typename _Tp >`
`void __valarray_fill (_Array< _Tp > __a, _Array< size_t > __i, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void __valarray_fill_construct (_Tp *__b, _Tp *__e, const _Tp __t)`
- `void *__valarray_get_memory (size_t __n)`
- `template<typename _Tp >`
`_Tp *__restrict __valarray_get_storage (size_t __n)`
- `template<typename _Ta >`
`_Ta::value_type __valarray_max (const _Ta &__a)`
- `template<typename _Ta >`
`_Ta::value_type __valarray_min (const _Ta &__a)`
- `template<typename _Tp >`
`_Tp __valarray_product (const _Tp *__f, const _Tp *__l)`
- `void __valarray_release_memory (void *__p)`
- `template<typename _Tp >`
`_Tp __valarray_sum (const _Tp *__f, const _Tp *__l)`
- `bool __verify_grouping (const char *__grouping, size_t __grouping_size, const string &__grouping_tmp) throw ()`
- `template<std::size_t _Ind, typename... _Tp>`
`auto __volget (volatile tuple< _Tp... > &__tuple) -> __tuple_element_t< _Ind, tuple< _Tp... >> volatile &`
- `template<std::size_t _Ind, typename... _Tp>`
`auto __volget (const volatile tuple< _Tp... > &__tuple) -> __tuple_element_t< _Ind, tuple< _Tp... >> const volatile &`
- `template<typename _CharT >`
`ostreambuf_iterator< _CharT > __write (ostreambuf_iterator< _CharT > __s, const _CharT *__ws, int __len)`
- `template<typename _CharT, typename _OutIter >`
`_OutIter __write (_OutIter __s, const _CharT *__ws, int __len)`
- `template<typename _Tp, class _Dom >`
`void __Array_augmented__bitwise_and (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void __Array_augmented__bitwise_and (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void __Array_augmented__bitwise_and (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void __Array_augmented__bitwise_and (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`

- `template<typename _Tp >`
`void _Array_augmented__bitwise_and (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_and (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__bitwise_and (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_and (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_and (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_and (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_and (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__bitwise_and (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__bitwise_or (_Array< _Tp > __a, size_t __n, const _Tp &__t)`

- `template<typename _Tp, class _Dom >`
`void _Array_augmented_bitwise_xor (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented_bitwise_xor (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented_bitwise_xor (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void _Array_augmented_bitwise_xor (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented_bitwise_xor (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented_bitwise_xor (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented_bitwise_xor (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void _Array_augmented_bitwise_xor (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented_bitwise_xor (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented_bitwise_xor (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented_bitwise_xor (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void _Array_augmented_bitwise_xor (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented_divides (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented_divides (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented_divides (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented_divides (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void _Array_augmented_divides (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented_divides (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented_divides (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented_divides (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void _Array_augmented_divides (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`

- `template<typename _Tp >`
`void _Array_augmented__divides (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__divides (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__divides (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__minus (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__minus (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void _Array_augmented__minus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__minus (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__minus (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void _Array_augmented__minus (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__minus (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__minus (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__minus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void _Array_augmented__minus (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__minus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__minus (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__modulus (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__modulus (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__modulus (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void _Array_augmented__modulus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__modulus (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__modulus (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void _Array_augmented__modulus (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`

- `template<typename _Tp >`
`void _Array_augmented__modulus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t >`
`__i)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__modulus (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e,`
`size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__modulus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool >`
`__m)`
- `template<typename _Tp >`
`void _Array_augmented__modulus (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t`
`__n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__modulus (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp >`
`&__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__multiplies (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t`
`__n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__multiplies (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp >`
`&__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__multiplies (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void _Array_augmented__multiplies (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void _Array_augmented__multiplies (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__multiplies (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__multiplies (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void _Array_augmented__multiplies (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t >`
`__i)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__multiplies (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp >`
`&__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__multiplies (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool >`
`__m)`
- `template<typename _Tp >`
`void _Array_augmented__multiplies (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t`
`__n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__multiplies (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e,`
`size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__plus (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__plus (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e,`
`size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__plus (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp > &__e,`
`size_t __n)`

- `template<typename _Tp >`
`void _Array_augmented_plus (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void _Array_augmented_plus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented_plus (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented_plus (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void _Array_augmented_plus (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented_plus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented_plus (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented_plus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp >`
`void _Array_augmented_plus (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented_shift_left (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void _Array_augmented_shift_left (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented_shift_left (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented_shift_left (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented_shift_left (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented_shift_left (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented_shift_left (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void _Array_augmented_shift_left (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented_shift_left (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented_shift_left (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented_shift_left (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented_shift_left (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void _Array_augmented_shift_right (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`

- `template<typename _Tp, class _Dom >`
`void _Array_augmented__shift_right (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__shift_right (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__shift_right (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void _Array_augmented__shift_right (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__shift_right (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__shift_right (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void _Array_augmented__shift_right (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__shift_right (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__shift_right (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void _Array_augmented__shift_right (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void _Array_augmented__shift_right (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _T1, typename... _Args>`
`void _Construct (_T1 *__p, _Args &&...__args)`
- `template<typename _Tp >`
`void _Destroy (_Tp *__pointer)`
- `template<typename _ForwardIterator >`
`void _Destroy (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Allocator >`
`void _Destroy (_ForwardIterator __first, _ForwardIterator __last, _Allocator &__alloc)`
- `template<typename _ForwardIterator, typename _Tp >`
`void _Destroy (_ForwardIterator __first, _ForwardIterator __last, allocator< _Tp > &)`
- `size_t _Fnv_hash_bytes (const void *__ptr, size_t __len, size_t __seed)`
- `size_t _Hash_bytes (const void *__ptr, size_t __len, size_t __seed)`
- `unsigned int _Rb_tree_black_count (const _Rb_tree_node_base *__node, const _Rb_tree_node_base *__root) throw ()`
- `_Rb_tree_node_base * _Rb_tree_decrement (_Rb_tree_node_base *__x) throw ()`
- `const _Rb_tree_node_base * _Rb_tree_decrement (const _Rb_tree_node_base *__x) throw ()`
- `_Rb_tree_node_base * _Rb_tree_increment (_Rb_tree_node_base *__x) throw ()`
- `const _Rb_tree_node_base * _Rb_tree_increment (const _Rb_tree_node_base *__x) throw ()`
- `void _Rb_tree_insert_and_rebalance (const bool __insert_left, _Rb_tree_node_base *__x, _Rb_tree_node_base *__p, _Rb_tree_node_base &__header) throw ()`
- `_Rb_tree_node_base * _Rb_tree_rebalance_for_erase (_Rb_tree_node_base *const __z, _Rb_tree_node_base &__header) throw ()`
- `void abort (void) throw ()`

- `template<typename _Tp >`
`_Tp abs (const complex< _Tp > &)`
- `constexpr double abs (double __x)`
- `constexpr float abs (float __x)`
- `constexpr long double abs (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type abs (_Tp __x)`
- `template<class _Dom >`
`_Expr< _UnClos< _Abs, _Expr, _Dom >, typename _Dom::value_type > abs (const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Abs, _ValArray, _Tp >, _Tp > abs (const valarray< _Tp > &__v)`
- `template<typename _InputIterator, typename _Tp >`
`_Tp accumulate (_InputIterator __first, _InputIterator __last, _Tp __init)`
- `template<typename _InputIterator, typename _Tp, typename _BinaryOperation >`
`_Tp accumulate (_InputIterator __first, _InputIterator __last, _Tp __init, _BinaryOperation __binary_op)`
- `constexpr float acos (float __x)`
- `constexpr long double acos (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type acos (_Tp __x)`
- `template<class _Dom >`
`_Expr< _UnClos< _Acos, _Expr, _Dom >, typename _Dom::value_type > acos (const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Acos, _ValArray, _Tp >, _Tp > acos (const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`std::complex< _Tp > acos (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > acosh (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`_Tp * addressof (_Tp &__r) noexcept`
- `template<typename _InputIterator, typename _OutputIterator >`
`_OutputIterator adjacent_difference (_InputIterator __first, _InputIterator __last, _OutputIterator __result)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator adjacent_difference (_InputIterator __first, _InputIterator __last, _OutputIterator __result, __ BinaryOperation __binary_op)`
- `template<typename _Filter >`
`_Filter adjacent_find (_Filter, _Filter)`
- `template<typename _Filter, typename _BinaryPredicate >`
`_Filter adjacent_find (_Filter, _Filter, _BinaryPredicate)`
- `template<typename _ForwardIterator >`
`_ForwardIterator adjacent_find (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _BinaryPredicate >`
`_ForwardIterator adjacent_find (_ForwardIterator __first, _ForwardIterator __last, _BinaryPredicate __binary_ ↵ pred)`
- `template<typename _InputIterator, typename _Distance >`
`void advance (_InputIterator &__i, _Distance __n)`
- `void * align (size_t __align, size_t __size, void *&__ptr, size_t &__space) noexcept`
- `template<typename _Iter, typename _Predicate >`
`bool all_of (_Iter, _Iter, _Predicate)`
- `template<typename _InputIterator, typename _Predicate >`
`bool all_of (_InputIterator __first, _InputIterator __last, _Predicate __pred)`

- `template<typename _Tp, typename _Alloc, typename... _Args>`
`shared_ptr< _Tp > allocate_shared (const _Alloc &__a, _Args &&... __args)`
- `template<typename _Iter, typename _Predicate >`
`bool any_of (_Iter, _Iter, _Predicate)`
- `template<typename _InputIterator, typename _Predicate >`
`bool any_of (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _Tp >`
`_Tp arg (const complex< _Tp > &)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type arg (_Tp __x)`
- `constexpr float asin (float __x)`
- `constexpr long double asin (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type asin (_Tp __x)`
- `template<class _Dom >`
`_Expr< _UnClos< _Asin, _Expr, _Dom >, typename _Dom::value_type > asin (const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Asin, _ValArray, _Tp >, _Tp > asin (const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`std::complex< _Tp > asin (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > asinh (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type assoc_laguerre (unsigned int __n, unsigned int __m, _Tp __x)`
- `float assoc_laguerref (unsigned int __n, unsigned int __m, float __x)`
- `long double assoc_laguerrel (unsigned int __n, unsigned int __m, long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type assoc_legendre (unsigned int __l, unsigned int __m, _Tp __x)`
- `float assoc_legendref (unsigned int __l, unsigned int __m, float __x)`
- `long double assoc_legendrel (unsigned int __l, unsigned int __m, long double __x)`
- `template<typename _Fn, typename... _Args>`
`future< __async_result_of< _Fn, _Args... > > async (launch __policy, _Fn &&__fn, _Args &&... __args)`
- `template<typename _Fn, typename... _Args>`
`future< __async_result_of< _Fn, _Args... > > async (_Fn &&__fn, _Args &&... __args)`
- `constexpr float atan (float __x)`
- `constexpr long double atan (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type atan (_Tp __x)`
- `template<class _Dom >`
`_Expr< _UnClos< _Atan, _Expr, _Dom >, typename _Dom::value_type > atan (const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Atan, _ValArray, _Tp >, _Tp > atan (const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`std::complex< _Tp > atan (const std::complex< _Tp > &__z)`
- `constexpr float atan2 (float __y, float __x)`
- `constexpr long double atan2 (long double __y, long double __x)`
- `template<typename _Tp, typename _Up >`
`constexpr __gnu_cxx::__promote_2< _Tp, _Up >::__type atan2 (_Tp __y, _Up __x)`
- `template<typename _Tp >`
`_Expr< _BinClos< _Atan2, _Constant, _ValArray, _Tp, _Tp >, _Tp > atan2 (const _Tp &__t, const valarray< _Tp > &__v)`

- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< _Atan2, _Expr, _Expr, _Dom1, _Dom2 >, typename _Dom1::value_type > atan2 (const _Expr< _Dom1, typename _Dom1::value_type > &__e1, const _Expr< _Dom2, typename _Dom2::value_type > &__e2)`
- `template<class _Dom >`
`_Expr< _BinClos< _Atan2, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename _Dom::value_type > atan2 (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< _Atan2, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_type > atan2 (const valarray< typename _Dom::valarray > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< _Atan2, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename _Dom::value_type > atan2 (const _Expr< _Dom, typename _Dom::value_type > &__e, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< _Atan2, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_type > atan2 (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _BinClos< _Atan2, _ValArray, _ValArray, _Tp, _Tp >, _Tp > atan2 (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< _Atan2, _ValArray, _Constant, _Tp, _Tp >, _Tp > atan2 (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`std::complex< _Tp > atanh (const std::complex< _Tp > &__z)`
- `int atexit (void (*)(void)) throw ()`
- `template<typename _ITp >`
`bool atomic_compare_exchange_strong (atomic< _ITp > *__a, _ITp *__i1, _ITp __i2) noexcept`
- `template<typename _ITp >`
`bool atomic_compare_exchange_strong (volatile atomic< _ITp > *__a, _ITp *__i1, _ITp __i2) noexcept`
- `template<typename _ITp >`
`bool atomic_compare_exchange_strong_explicit (atomic< _ITp > *__a, _ITp *__i1, _ITp __i2, memory_order __m1, memory_order __m2) noexcept`
- `template<typename _ITp >`
`bool atomic_compare_exchange_strong_explicit (volatile atomic< _ITp > *__a, _ITp *__i1, _ITp __i2, memory_order __m1, memory_order __m2) noexcept`
- `template<typename _ITp >`
`bool atomic_compare_exchange_weak (atomic< _ITp > *__a, _ITp *__i1, _ITp __i2) noexcept`
- `template<typename _ITp >`
`bool atomic_compare_exchange_weak (volatile atomic< _ITp > *__a, _ITp *__i1, _ITp __i2) noexcept`
- `template<typename _ITp >`
`bool atomic_compare_exchange_weak_explicit (atomic< _ITp > *__a, _ITp *__i1, _ITp __i2, memory_order __m1, memory_order __m2) noexcept`
- `template<typename _ITp >`
`bool atomic_compare_exchange_weak_explicit (volatile atomic< _ITp > *__a, _ITp *__i1, _ITp __i2, memory_order __m1, memory_order __m2) noexcept`
- `template<typename _ITp >`
`_ITp atomic_exchange (atomic< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp atomic_exchange (volatile atomic< _ITp > *__a, _ITp __i) noexcept`

- `template<typename _ITp >`
`_ITp atomic_exchange_explicit (atomic<_ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_exchange_explicit (volatile atomic<_ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_add (__atomic_base<_ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_add (volatile __atomic_base<_ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp * atomic_fetch_add (volatile atomic<_ITp * > *__a, ptrdiff_t __d) noexcept`
- `template<typename _ITp >`
`_ITp * atomic_fetch_add (atomic<_ITp * > *__a, ptrdiff_t __d) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_add_explicit (__atomic_base<_ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_add_explicit (volatile __atomic_base<_ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp * atomic_fetch_add_explicit (atomic<_ITp * > *__a, ptrdiff_t __d, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp * atomic_fetch_add_explicit (volatile atomic<_ITp * > *__a, ptrdiff_t __d, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_and (__atomic_base<_ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_and (volatile __atomic_base<_ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_and_explicit (__atomic_base<_ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_and_explicit (volatile __atomic_base<_ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_or (__atomic_base<_ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_or (volatile __atomic_base<_ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_or_explicit (__atomic_base<_ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_or_explicit (volatile __atomic_base<_ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_sub (__atomic_base<_ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_sub (volatile __atomic_base<_ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp * atomic_fetch_sub (volatile atomic<_ITp * > *__a, ptrdiff_t __d) noexcept`
- `template<typename _ITp >`
`_ITp * atomic_fetch_sub (atomic<_ITp * > *__a, ptrdiff_t __d) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_sub_explicit (__atomic_base<_ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_sub_explicit (volatile __atomic_base<_ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp * atomic_fetch_sub_explicit (volatile atomic<_ITp * > *__a, ptrdiff_t __d, memory_order __m) noexcept`

- `template<typename _ITp >`
`_ITp * atomic_fetch_sub_explicit (atomic<_ITp > *__a, ptrdiff_t __d, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_xor (__atomic_base<_ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_xor (volatile __atomic_base<_ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_xor_explicit (__atomic_base<_ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_fetch_xor_explicit (volatile __atomic_base<_ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `void atomic_flag_clear (atomic_flag *__a) noexcept`
- `void atomic_flag_clear (volatile atomic_flag *__a) noexcept`
- `void atomic_flag_clear_explicit (atomic_flag *__a, memory_order __m) noexcept`
- `void atomic_flag_clear_explicit (volatile atomic_flag *__a, memory_order __m) noexcept`
- `bool atomic_flag_test_and_set (atomic_flag *__a) noexcept`
- `bool atomic_flag_test_and_set (volatile atomic_flag *__a) noexcept`
- `bool atomic_flag_test_and_set_explicit (atomic_flag *__a, memory_order __m) noexcept`
- `bool atomic_flag_test_and_set_explicit (volatile atomic_flag *__a, memory_order __m) noexcept`
- `template<typename _ITp >`
`void atomic_init (atomic<_ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`void atomic_init (volatile atomic<_ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`bool atomic_is_lock_free (const atomic<_ITp > *__a) noexcept`
- `template<typename _ITp >`
`bool atomic_is_lock_free (const volatile atomic<_ITp > *__a) noexcept`
- `template<typename _ITp >`
`_ITp atomic_load (const atomic<_ITp > *__a) noexcept`
- `template<typename _ITp >`
`_ITp atomic_load (const volatile atomic<_ITp > *__a) noexcept`
- `template<typename _ITp >`
`_ITp atomic_load_explicit (const atomic<_ITp > *__a, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp atomic_load_explicit (const volatile atomic<_ITp > *__a, memory_order __m) noexcept`
- `template<typename _ITp >`
`void atomic_store (atomic<_ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`void atomic_store (volatile atomic<_ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`void atomic_store_explicit (atomic<_ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`void atomic_store_explicit (volatile atomic<_ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _Container >`
`back_insert_iterator<_Container > back_inserter (_Container &__x)`
- `template<typename _Container >`
`auto begin (_Container &__cont) -> decltype(__cont.begin())`
- `template<typename _Container >`
`auto begin (const _Container &__cont) -> decltype(__cont.begin())`
- `template<typename _Tp, size_t _Nm>`
`_GLIBCXX14_CONSTEXPR _Tp * begin (_Tp(&__arr)[_Nm])`
- `template<class _Tp >`
`constexpr const _Tp * begin (initializer_list<_Tp > __ils) noexcept`

- `template<class _Tp >`
`_Tp * begin (valarray< _Tp > &__va)`
- `template<class _Tp >`
`const _Tp * begin (const valarray< _Tp > &__va)`
- `template<typename _Tpa, typename _Tpb >`
`__gnu_cxx::__promote_2< _Tpa, _Tpb >::__type beta (_Tpa __a, _Tpb __b)`
- `float betaf (float __a, float __b)`
- `long double betal (long double __a, long double __b)`
- `template<typename _Filter, typename _Tp >`
`bool binary_search (_Filter, _Filter, const _Tp &)`
- `template<typename _Filter, typename _Tp, typename _Compare >`
`bool binary_search (_Filter, _Filter, const _Tp &, _Compare)`
- `template<typename _ForwardIterator, typename _Tp >`
`bool binary_search (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __val)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`bool binary_search (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __val, _Compare __comp)`
- `template<typename _Func, typename... _BoundArgs>`
`_Bind_helper< __is_socketlike< _Func >::value, _Func, _BoundArgs... >::type bind (_Func && __f, _BoundArgs &&... __args)`
- `template<typename _Result, typename _Func, typename... _BoundArgs>`
`_Bindres_helper< _Result, _Func, _BoundArgs... >::type bind (_Func && __f, _BoundArgs &&... __args)`
- `template<typename _Operation, typename _Tp >`
`binder1st< _Operation > bind1st (const _Operation & __fn, const _Tp & __x)`
- `template<typename _Operation, typename _Tp >`
`binder2nd< _Operation > bind2nd (const _Operation & __fn, const _Tp & __x)`
- `ios_base & boolalpha (ios_base & __base)`
- `template<typename _Container >`
`constexpr auto cbegin (const _Container & __cont) noexcept(noexcept(std::begin(__cont))) -> decltype(std::begin(__cont))`
- `constexpr float ceil (float __x)`
- `constexpr long double ceil (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type ceil (_Tp __x)`
- `template<typename _Container >`
`constexpr auto cend (const _Container & __cont) noexcept(noexcept(std::end(__cont))) -> decltype(std::end(__cont))`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type comp_ellint_1 (_Tp __k)`
- `float comp_ellint_1f (float __k)`
- `long double comp_ellint_1l (long double __k)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type comp_ellint_2 (_Tp __k)`
- `float comp_ellint_2f (float __k)`
- `long double comp_ellint_2l (long double __k)`
- `template<typename _Tp, typename _Tpn >`
`__gnu_cxx::__promote_2< _Tp, _Tpn >::__type comp_ellint_3 (_Tp __k, _Tpn __nu)`
- `float comp_ellint_3f (float __k, float __nu)`
- `long double comp_ellint_3l (long double __k, long double __nu)`
- `template<typename _Tp >`
`complex< _Tp > conj (const complex< _Tp > &)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type conj (_Tp __x)`

- `template<typename _Tp, typename _Tp1 >`
`shared_ptr< _Tp > const_pointer_cast (const shared_ptr< _Tp1 > &__r) noexcept`
- `template<typename _Tp, typename _Tp1, _Lock_policy _Lp>`
`__shared_ptr< _Tp, _Lp > const_pointer_cast (const __shared_ptr< _Tp1, _Lp > &__r) noexcept`
- `template<typename _Iter, typename _OIter >`
`_OIter copy (_Iter, _Iter, _OIter)`
- `template<typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, ostreambuf_iterator< _CharT >::__type >::type copy (istreambuf_iterator< _CharT > __first, istreambuf_iterator< _CharT > __last, ostreambuf_iterator< _CharT > __result)`
- `template<typename _Tp >`
`_Deque_iterator< _Tp, _Tp &, _Tp * > copy (_Deque_iterator< _Tp, _Tp &, _Tp * > __first, _Deque_iterator< _Tp, _Tp &, _Tp * > __last, _Deque_iterator< _Tp, _Tp &, _Tp * > __result)`
- `template<typename _II, typename _OI >`
`_OI copy (_II __first, _II __last, _OI __result)`
- `template<typename _Tp >`
`_Deque_iterator< _Tp, _Tp &, _Tp * > copy (_Deque_iterator< _Tp, const _Tp &, const _Tp * > __first, ↵
_Deque_iterator< _Tp, const _Tp &, const _Tp * > __last, _Deque_iterator< _Tp, _Tp &, _Tp * > __result)`
- `template<typename _Blter1, typename _Blter2 >`
`_Blter2 copy_backward (_Blter1, _Blter1, _Blter2)`
- `template<typename _Tp >`
`_Deque_iterator< _Tp, _Tp &, _Tp * > copy_backward (_Deque_iterator< _Tp, _Tp &, _Tp * > __first, ↵
_Deque_iterator< _Tp, _Tp &, _Tp * > __last, _Deque_iterator< _Tp, _Tp &, _Tp * > __result)`
- `template<typename _BI1, typename _BI2 >`
`_BI2 copy_backward (_BI1 __first, _BI1 __last, _BI2 __result)`
- `template<typename _Tp >`
`_Deque_iterator< _Tp, _Tp &, _Tp * > copy_backward (_Deque_iterator< _Tp, const _Tp &, const _Tp * > __first, ↵
_Deque_iterator< _Tp, const _Tp &, const _Tp * > __last, _Deque_iterator< _Tp, _Tp &, _Tp * > __result)`
- `template<typename _Ex >`
`exception_ptr copy_exception (_Ex __ex) noexcept 1`
- `template<typename _Iter, typename _OIter, typename _Predicate >`
`_OIter copy_if (_Iter, _Iter, _OIter, _Predicate)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate >`
`_OutputIterator copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Predicate __pred)`
- `template<typename _Iter, typename _Size, typename _OIter >`
`_OIter copy_n (_Iter, _Size, _OIter)`
- `template<typename _InputIterator, typename _Size, typename _OutputIterator >`
`_OutputIterator copy_n (_InputIterator __first, _Size __n, _OutputIterator __result)`
- `template<typename _Tp >`
`complex< _Tp > cos (const complex< _Tp > &)`
- `constexpr float cos (float __x)`
- `constexpr long double cos (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type cos (_Tp __x)`
- `template<class _Dom >`
`_Expr< _UnClos< _Cos, _Expr, _Dom >, typename _Dom::value_type > cos (const _Expr< _Dom, typename
_Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Cos, _ValArray, _Tp >, _Tp > cos (const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`complex< _Tp > cosh (const complex< _Tp > &)`
- `constexpr float cosh (float __x)`

- constexpr long double **cosh** (long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **cosh** (_Tp __x)
- template<typename _Tp >
_Expr< _UnClos< _Cosh, _ValArray, _Tp >, _Tp > **cosh** (const valarray< _Tp > &__v)
- template<class _Dom >
_Expr< _UnClos< _Cosh, _Expr, _Dom >, typename _Dom::value_type > **cosh** (const _Expr< _Dom, typename _Dom::value_type > &__e)
- template<typename _Iter, typename _Tp >
iterator_traits< _Iter >::difference_type **count** (_Iter, _Iter, const _Tp &)
- template<typename _InputIterator, typename _Tp >
iterator_traits< _InputIterator >::difference_type **count** (_InputIterator __first, _InputIterator __last, const _Tp & __value)
- template<typename _Iter, typename _Predicate >
iterator_traits< _Iter >::difference_type **count_if** (_Iter, _Iter, _Predicate)
- template<typename _InputIterator, typename _Predicate >
iterator_traits< _InputIterator >::difference_type **count_if** (_InputIterator __first, _InputIterator __last, _Predicate __pred)
- template<typename _Container >
auto **crbegin** (const _Container &__cont) -> decltype(std::rbegin(__cont))
- template<typename _Container >
auto **crend** (const _Container &__cont) -> decltype(std::rend(__cont))
- exception_ptr **current_exception** () noexcept
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type **cyl_bessel_i** (_Tpnu __nu, _Tp __x)
- float **cyl_bessel_if** (float __nu, float __x)
- long double **cyl_bessel_il** (long double __nu, long double __x)
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type **cyl_bessel_j** (_Tpnu __nu, _Tp __x)
- float **cyl_bessel_jf** (float __nu, float __x)
- long double **cyl_bessel_jl** (long double __nu, long double __x)
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type **cyl_bessel_k** (_Tpnu __nu, _Tp __x)
- float **cyl_bessel_kf** (float __nu, float __x)
- long double **cyl_bessel_kl** (long double __nu, long double __x)
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type **cyl_neumann** (_Tpnu __nu, _Tp __x)
- float **cyl_neumannf** (float __nu, float __x)
- long double **cyl_neumannl** (long double __nu, long double __x)
- **ios_base** & **dec** (**ios_base** & __base)
- void **declare_no_pointers** (char *, size_t)
- void **declare_reachable** (void *)
- **ios_base** & **defaultfloat** (**ios_base** & __base)
- template<typename _InputIterator >
iterator_traits< _InputIterator >::difference_type **distance** (_InputIterator __first, _InputIterator __last)
- template<typename _Tp, typename _Tp1 >
shared_ptr< _Tp > **dynamic_pointer_cast** (const **shared_ptr**< _Tp1 > &__r) noexcept
- template<typename _Tp, typename _Tp1, _Lock_policy _Lp>
__shared_ptr< _Tp, _Lp > **dynamic_pointer_cast** (const __shared_ptr< _Tp1, _Lp > &__r) noexcept
- template<typename _Tp, typename _Tpp >
__gnu_cxx::__promote_2< _Tp, _Tpp >::__type **ellint_1** (_Tp __k, _Tpp __phi)
- float **ellint_1f** (float __k, float __phi)

- long double [ellint_1l](#) (long double __k, long double __phi)
- template<typename _Tp, typename _Tpp >
__gnu_cxx::__promote_2< _Tp, _Tpp >::__type [ellint_2](#) (_Tp __k, _Tpp __phi)
- float [ellint_2f](#) (float __k, float __phi)
- long double [ellint_2l](#) (long double __k, long double __phi)
- template<typename _Tp, typename _Tpn, typename _Tpp >
__gnu_cxx::__promote_3< _Tp, _Tpn, _Tpp >::__type [ellint_3](#) (_Tp __k, _Tpn __nu, _Tpp __phi)
- float [ellint_3f](#) (float __k, float __nu, float __phi)
- long double [ellint_3l](#) (long double __k, long double __nu, long double __phi)
- template<typename _Container >
auto [end](#) (_Container &__cont) -> decltype(__cont.end())
- template<typename _Container >
auto [end](#) (const _Container &__cont) -> decltype(__cont.end())
- template<typename _Tp, size_t _Nm>
__GLIBCXX14_CONSTEXPR _Tp * [end](#) (_Tp(&__arr)[_Nm])
- template<class _Tp >
constexpr const _Tp * [end](#) (initializer_list< _Tp > __ils) noexcept
- template<class _Tp >
_Tp * [end](#) (valarray< _Tp > &__va)
- template<class _Tp >
const _Tp * [end](#) (const valarray< _Tp > &__va)
- template<typename _CharT, typename _Traits >
[basic_ostream](#)< _CharT, _Traits > & [endl](#) ([basic_ostream](#)< _CharT, _Traits > &__os)
- template<typename _CharT, typename _Traits >
[basic_ostream](#)< _CharT, _Traits > & [ends](#) ([basic_ostream](#)< _CharT, _Traits > &__os)
- template<typename _Iter1, typename _Iter2 >
bool [equal](#) (_Iter1, _Iter1, _Iter2)
- template<typename _Iter1, typename _Iter2, typename _BinaryPredicate >
bool [equal](#) (_Iter1 __first1, _Iter1 __last1, _Iter2 __first2, _BinaryPredicate __binary_pred)
- template<typename _II1, typename _II2 >
bool [equal](#) (_II1 __first1, _II1 __last1, _II2 __first2)
- template<typename _II1, typename _II2 >
bool [equal](#) (_II1 __first1, _II1 __last1, _II2 __first2, _II2 __last2)
- template<typename _Iter1, typename _Iter2, typename _BinaryPredicate >
bool [equal](#) (_Iter1 __first1, _Iter1 __last1, _Iter2 __first2, _Iter2 __last2, _BinaryPredicate __binary_pred)
- template<typename _Filter, typename _Tp >
[pair](#)< _Filter, _Filter > [equal_range](#) (_Filter, _Filter, const _Tp &)
- template<typename _Filter, typename _Tp, typename _Compare >
[pair](#)< _Filter, _Filter > [equal_range](#) (_Filter, _Filter, const _Tp &, _Compare)
- template<typename _ForwardIterator, typename _Tp >
[pair](#)< _ForwardIterator, _ForwardIterator > [equal_range](#) (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __val)
- template<typename _ForwardIterator, typename _Tp, typename _Compare >
[pair](#)< _ForwardIterator, _ForwardIterator > [equal_range](#) (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __val, _Compare __comp)
- template<typename _Tp, typename _Up = _Tp>
_Tp [exchange](#) (_Tp &__obj, _Up &&__new_val)
- void [exit](#) (int) throw ()
- template<typename _Tp >
[complex](#)< _Tp > [exp](#) (const [complex](#)< _Tp > &)
- constexpr float [exp](#) (float __x)
- constexpr long double [exp](#) (long double __x)

- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type exp (_Tp __x)`
- `template<class _Dom >`
`_Expr< _UnClos< _Exp, _Expr, _Dom >, typename _Dom::value_type > exp (const _Expr< _Dom, typename`
`_Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Exp, _ValArray, _Tp >, _Tp > exp (const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type expint (_Tp __x)`
- `float expintf (float __x)`
- `long double expintl (long double __x)`
- `constexpr float fabs (float __x)`
- `constexpr long double fabs (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type fabs (_Tp __x)`
- `template<typename _Tp >`
`_Tp fabs (const std::complex< _Tp > &__z)`
- `template<typename _Filter, typename _Tp >`
`void fill (_Filter, _Filter, const _Tp &)`
- `void fill (_Bit_iterator __first, _Bit_iterator __last, const bool &__x)`
- `template<typename _ForwardIterator, typename _Tp >`
`void fill (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value)`
- `template<typename _Tp >`
`void fill (const Deque_iterator< _Tp, _Tp &, _Tp * > &__first, const Deque_iterator< _Tp, _Tp &, _Tp * >`
`&__last, const _Tp &__value)`
- `template<typename _OIter, typename _Size, typename _Tp >`
`_OIter fill_n (_OIter, _Size, const _Tp &)`
- `template<typename _OI, typename _Size, typename _Tp >`
`_OI fill_n (_OI __first, _Size __n, const _Tp &__value)`
- `template<typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, istreambuf_iterator< _CharT > >::__type find`
`(istreambuf_iterator< _CharT > __first, istreambuf_iterator< _CharT > __last, const _CharT &__val)`
- `template<typename _Iter, typename _Tp >`
`_Iter find (_Iter, _Iter, const _Tp &)`
- `template<typename _InputIterator, typename _Tp >`
`_InputIterator find (_InputIterator __first, _InputIterator __last, const _Tp &__val)`
- `template<typename _Filter1, typename _Filter2 >`
`_Filter1 find_end (_Filter1, _Filter1, _Filter2, _Filter2)`
- `template<typename _Filter1, typename _Filter2, typename _BinaryPredicate >`
`_Filter1 find_end (_Filter1, _Filter1, _Filter2, _Filter2, _BinaryPredicate)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`_ForwardIterator1 find_end (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, ↵`
`_ForwardIterator2 __last2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`_ForwardIterator1 find_end (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, ↵`
`_ForwardIterator2 __last2, _BinaryPredicate __comp)`
- `template<typename _Filter1, typename _Filter2 >`
`_Filter1 find_first_of (_Filter1, _Filter1, _Filter2, _Filter2)`
- `template<typename _Filter1, typename _Filter2, typename _BinaryPredicate >`
`_Filter1 find_first_of (_Filter1, _Filter1, _Filter2, _Filter2, _BinaryPredicate)`
- `template<typename _InputIterator, typename _ForwardIterator >`
`_InputIterator find_first_of (_InputIterator __first1, _InputIterator __last1, _ForwardIterator __first2, _Forward↵`
`Iterator __last2)`

- `template<typename _InputIterator, typename _ForwardIterator, typename _BinaryPredicate >`
`_InputIterator find_first_of (_InputIterator __first1, _InputIterator __last1, _ForwardIterator __first2, _ForwardIterator __last2, _BinaryPredicate __comp)`
- `template<typename _Iter, typename _Predicate >`
`_Iter find_if (_Iter, _Iter, _Predicate)`
- `template<typename _InputIterator, typename _Predicate >`
`_InputIterator find_if (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _Iter, typename _Predicate >`
`_Iter find_if_not (_Iter, _Iter, _Predicate)`
- `template<typename _InputIterator, typename _Predicate >`
`_InputIterator find_if_not (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `ios_base & fixed (ios_base &__base)`
- `constexpr float floor (float __x)`
- `constexpr long double floor (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type floor (_Tp __x)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & flush (basic_ostream< _CharT, _Traits > &__os)`
- `constexpr float fmod (float __x, float __y)`
- `constexpr long double fmod (long double __x, long double __y)`
- `template<typename _Tp, typename _Up >`
`constexpr __gnu_cxx::__promote_2< _Tp, _Up >::__type fmod (_Tp __x, _Up __y)`
- `template<typename _Iter, typename _Funct >`
`_Funct for_each (_Iter, _Iter, _Funct)`
- `template<typename _InputIterator, typename _Function >`
`_Function for_each (_InputIterator __first, _InputIterator __last, _Function __f)`
- `template<typename _Tp >`
`constexpr _Tp && forward (typename std::remove_reference< _Tp >::type &__t) noexcept`
- `template<typename _Tp >`
`constexpr _Tp && forward (typename std::remove_reference< _Tp >::type &&__t) noexcept`
- `template<typename... _Elements>`
`constexpr tuple< _Elements &&... > forward_as_tuple (_Elements &&... __args) noexcept`
- `float frexp (float __x, int *__exp)`
- `long double frexp (long double __x, int *__exp)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type frexp (_Tp __x, int *__exp)`
- `template<typename _Container >`
`front_insert_iterator< _Container > front_inserter (_Container &__x)`
- `const error_category & future_category () noexcept`
- `template<typename _Filter, typename _Generator >`
`void generate (_Filter, _Filter, _Generator)`
- `template<typename _ForwardIterator, typename _Generator >`
`void generate (_ForwardIterator __first, _ForwardIterator __last, _Generator __gen)`
- `template<typename _RealType, size_t __bits, typename _UniformRandomNumberGenerator >`
`_RealType generate_canonical (_UniformRandomNumberGenerator &__g)`
- `template<typename _OIter, typename _Size, typename _Generator >`
`_OIter generate_n (_OIter, _Size, _Generator)`
- `template<typename _OutputIterator, typename _Size, typename _Generator >`
`_OutputIterator generate_n (_OutputIterator __first, _Size __n, _Generator __gen)`
- `template<std::size_t __Int, class _Tp1, class _Tp2 >`
`constexpr tuple_element< __Int, std::pair< _Tp1, _Tp2 > >::type & get (std::pair< _Tp1, _Tp2 > &__in) noexcept`

- `template<std::size_t _Int, class _Tp1, class _Tp2 >`
`constexpr tuple_element< _Int, std::pair< _Tp1, _Tp2 > >::type && get (std::pair< _Tp1, _Tp2 > &&__in)`
`noexcept`
- `template<std::size_t _Int, class _Tp1, class _Tp2 >`
`constexpr const tuple_element< _Int, std::pair< _Tp1, _Tp2 > >::type & get (const std::pair< _Tp1, _Tp2 >`
`&__in) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr _Tp & get (pair< _Tp, _Up > &__p) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr const _Tp & get (const pair< _Tp, _Up > &__p) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr _Tp && get (pair< _Tp, _Up > &&__p) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr _Tp & get (pair< _Up, _Tp > &__p) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr const _Tp & get (const pair< _Up, _Tp > &__p) noexcept`
- `template<typename _Tp, typename _Up >`
`constexpr _Tp && get (pair< _Up, _Tp > &&__p) noexcept`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>`
`constexpr _Tp & get (array< _Tp, _Nm > &__arr) noexcept`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>`
`constexpr _Tp && get (array< _Tp, _Nm > &&__arr) noexcept`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>`
`constexpr const _Tp & get (const array< _Tp, _Nm > &__arr) noexcept`
- `template<std::size_t __i, typename... _Elements>`
`constexpr __tuple_element_t< __i, tuple< _Elements... > > & get (tuple< _Elements... > &__t) noexcept`
- `template<std::size_t __i, typename... _Elements>`
`constexpr const __tuple_element_t< __i, tuple< _Elements... > > & get (const tuple< _Elements... > &__t)`
`noexcept`
- `template<std::size_t __i, typename... _Elements>`
`constexpr __tuple_element_t< __i, tuple< _Elements... > > && get (tuple< _Elements... > &&__t) noexcept`
- `template<typename _Tp, typename... _Types>`
`constexpr _Tp & get (tuple< _Types... > &__t) noexcept`
- `template<typename _Tp, typename... _Types>`
`constexpr _Tp && get (tuple< _Types... > &&__t) noexcept`
- `template<typename _Tp, typename... _Types>`
`constexpr const _Tp & get (const tuple< _Types... > &__t) noexcept`
- **Catalogs & get_catalogs ()**
- `template<typename _Del, typename _Tp, _Lock_policy _Lp>`
`_Del * get_deleter (const __shared_ptr< _Tp, _Lp > &__p) noexcept`
- `template<typename _MoneyT >`
`_Get_money< _MoneyT > get_money (_MoneyT &__mon, bool __intl=false)`
- **new_handler get_new_handler ()** noexcept
- **pointer_safety get_pointer_safety ()** noexcept
- `template<typename _Tp >`
`pair< _Tp *, ptrdiff_t > get_temporary_buffer (ptrdiff_t __len) noexcept`
- **terminate_handler get_terminate ()** noexcept
- `template<typename _CharT >`
`_Get_time< _CharT > get_time (std::tm *__tmb, const _CharT *__fmt)`
- **unexpected_handler get_unexpected ()** noexcept
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`basic_istream< _CharT, _Traits > & getline (basic_istream< _CharT, _Traits > &__is, __gnu_cxx::__versa_↵`
`string< _CharT, _Traits, _Alloc, _Base > &__str, _CharT __delim)`

- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>
basic_istream< _CharT, _Traits > & getline (basic_istream< _CharT, _Traits > &__is, __gnu_cxx::__versa↵
string< _CharT, _Traits, _Alloc, _Base > &__str)`
- `template<typename _CharT, typename _Traits, typename _Alloc >
basic_istream< _CharT, _Traits > & getline (basic_istream< _CharT, _Traits > &__is, basic_string< _CharT,
_Traits, _Alloc > &__str, _CharT __delim)`
- `template<typename _CharT, typename _Traits, typename _Alloc >
basic_istream< _CharT, _Traits > & getline (basic_istream< _CharT, _Traits > &__is, basic_string< _CharT,
_Traits, _Alloc > &__str)`
- `template<typename _CharT, typename _Traits, typename _Alloc >
basic_istream< _CharT, _Traits > & getline (basic_istream< _CharT, _Traits > &&__is, basic_string< _CharT,
_Traits, _Alloc > &__str, _CharT __delim)`
- `template<typename _CharT, typename _Traits, typename _Alloc >
basic_istream< _CharT, _Traits > & getline (basic_istream< _CharT, _Traits > &&__is, basic_string< _CharT,
_Traits, _Alloc > &__str)`
- `template<>
basic_istream< char > & getline (basic_istream< char > &__in, basic_string< char > &__str, char __delim)`
- `template<>
basic_istream< wchar_t > & getline (basic_istream< wchar_t > &__in, basic_string< wchar_t > &__str,
wchar_t __delim)`
- `template<typename _Facet >
bool has_facet (const locale &__loc) throw ()`
- `template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type hermite (unsigned int __n, _Tp __x)`
- `float hermitef (unsigned int __n, float __x)`
- `long double hermitel (unsigned int __n, long double __x)`
- `ios_base & hex (ios_base &__base)`
- `ios_base & hexfloat (ios_base &__base)`
- `template<typename _Tp >
constexpr _Tp imag (const complex< _Tp > &__z)`
- `template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type imag (_Tp)`
- `template<typename _Iter1, typename _Iter2 >
bool includes (_Iter1, _Iter1, _Iter2, _Iter2)`
- `template<typename _Iter1, typename _Iter2, typename _Compare >
bool includes (_Iter1, _Iter1, _Iter2, _Iter2, _Compare)`
- `template<typename _InputIterator1, typename _InputIterator2 >
bool includes (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Compare >
bool includes (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2,
_Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Tp >
_Tp inner_product (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Tp __init)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Tp, typename _BinaryOperation1, typename _BinaryOperation2
>
_Tp inner_product (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Tp __init, _Binary↵
Operation1 __binary_op1, _BinaryOperation2 __binary_op2)`
- `template<typename _Biter >
void inplace_merge (_Biter, _Biter, _Biter)`
- `template<typename _Biter, typename _Compare >
void inplace_merge (_Biter, _Biter, _Biter, _Compare)`
- `template<typename _BidirectionalIterator >
void inplace_merge (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last)`

- `template<typename _BidirectionalIterator, typename _Compare >`
`void inplace_merge (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last, ↵`
`_Compare __comp)`
- `template<typename _Container, typename _Iterator >`
`insert_iterator< _Container > insertter (_Container &__x, _Iterator __i)`
- `ios_base & internal (ios_base &__base)`
- `const error_category & iostream_category () noexcept`
- `template<typename _ForwardIterator, typename _Tp >`
`void iota (_ForwardIterator __first, _ForwardIterator __last, _Tp __value)`
- `template<typename _RAIter >`
`bool is_heap (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`bool is_heap (_RAIter, _RAIter, _Compare)`
- `template<typename _RandomAccessIterator >`
`bool is_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`bool is_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RAIter >`
`_RAIter is_heap_until (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`_RAIter is_heap_until (_RAIter, _RAIter, _Compare)`
- `template<typename _RandomAccessIterator >`
`_RandomAccessIterator is_heap_until (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`_RandomAccessIterator is_heap_until (_RandomAccessIterator __first, _RandomAccessIterator __last, ↵`
`_Compare __comp)`
- `template<typename _Iter, typename _Predicate >`
`bool is_partitioned (_Iter, _Iter, _Predicate)`
- `template<typename _InputIterator, typename _Predicate >`
`bool is_partitioned (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _Filter1, typename _Filter2 >`
`bool is_permutation (_Filter1, _Filter1, _Filter2)`
- `template<typename _Filter1, typename _Filter2, typename _BinaryPredicate >`
`bool is_permutation (_Filter1, _Filter1, _Filter2, _BinaryPredicate)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`bool is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`bool is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, ↵`
`_BinaryPredicate __pred)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`bool is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, ↵`
`_ForwardIterator2 __last2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`bool is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, ↵`
`_ForwardIterator2 __last2, _BinaryPredicate __pred)`
- `template<typename _Filter >`
`bool is_sorted (_Filter, _Filter)`
- `template<typename _Filter, typename _Compare >`
`bool is_sorted (_Filter, _Filter, _Compare)`
- `template<typename _ForwardIterator >`
`bool is_sorted (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`bool is_sorted (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`

- `template<typename _Filter >`
`_Filter is_sorted_until (_Filter, _Filter)`
- `template<typename _Filter, typename _Compare >`
`_Filter is_sorted_until (_Filter, _Filter, _Compare)`
- `template<typename _ForwardIterator >`
`_ForwardIterator is_sorted_until (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`_ForwardIterator is_sorted_until (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _CharT >`
`bool isalnum (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool isalpha (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool isblank (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool iscntrl (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool isdigit (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool isgraph (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool islower (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool isprint (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool ispunct (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool isspace (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool isupper (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool isxdigit (_CharT __c, const locale &__loc)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`void iter_swap (_ForwardIterator1 __a, _ForwardIterator2 __b)`
- `template<typename _Filter1, typename _Filter2 >`
`void iter_swap (_Filter1, _Filter2)`
- `template<typename _Tp >`
`_Tp kill_dependency (_Tp __y) noexcept`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type laguerre (unsigned int __n, _Tp __x)`
- `float laguerref (unsigned int __n, float __x)`
- `long double laguerrel (unsigned int __n, long double __x)`
- `constexpr float ldexp (float __x, int __exp)`
- `constexpr long double ldexp (long double __x, int __exp)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type ldexp (_Tp __x, int __exp)`
- `ios_base & left (ios_base &__base)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type legendre (unsigned int __l, _Tp __x)`
- `float legendref (unsigned int __l, float __x)`
- `long double legendrel (unsigned int __l, long double __x)`

- `template<typename _Iter1, typename _Iter2 >`
`bool lexicographical_compare (_Iter1, _Iter1, _Iter2, _Iter2)`
- `template<typename _Iter1, typename _Iter2, typename _Compare >`
`bool lexicographical_compare (_Iter1, _Iter1, _Iter2, _Iter2, _Compare)`
- `template<typename _I1, typename _I2 >`
`bool lexicographical_compare (_I1 __first1, _I1 __last1, _I2 __first2, _I2 __last2)`
- `template<typename _I1, typename _I2, typename _Compare >`
`bool lexicographical_compare (_I1 __first1, _I1 __last1, _I2 __first2, _I2 __last2, _Compare __comp)`
- `template<typename _Tp >`
`complex< _Tp > log (const complex< _Tp > &)`
- `constexpr float log (float __x)`
- `constexpr long double log (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type log (_Tp __x)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Log, _ValArray, _Tp >, _Tp > log (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< _Log, _Expr, _Dom >, typename _Dom::value_type > log (const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`complex< _Tp > log10 (const complex< _Tp > &)`
- `constexpr float log10 (float __x)`
- `constexpr long double log10 (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type log10 (_Tp __x)`
- `template<class _Dom >`
`_Expr< _UnClos< _Log10, _Expr, _Dom >, typename _Dom::value_type > log10 (const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Log10, _ValArray, _Tp >, _Tp > log10 (const valarray< _Tp > &__v)`
- `template<typename _Filter, typename _Tp >`
`_Filter lower_bound (_Filter, _Filter, const _Tp &)`
- `template<typename _Filter, typename _Tp, typename _Compare >`
`_Filter lower_bound (_Filter, _Filter, const _Tp &, _Compare)`
- `template<typename _ForwardIterator, typename _Tp >`
`_ForwardIterator lower_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`_ForwardIterator lower_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)`
- `error_code make_error_code (future_errc __errc) noexcept`
- `error_code make_error_code (errc __e) noexcept`
- `error_code make_error_code (io_errc e) noexcept`
- `error_condition make_error_condition (future_errc __errc) noexcept`
- `error_condition make_error_condition (io_errc e) noexcept`
- `error_condition make_error_condition (errc __e) noexcept`
- `template<typename _Ex >`
`exception_ptr make_exception_ptr (_Ex __ex) noexcept`
- `template<typename _RAIter >`
`void make_heap (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`void make_heap (_RAIter, _RAIter, _Compare)`

- `template<typename _RandomAccessIterator >`
`void make_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void make_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _Iterator >`
`move_iterator< _Iterator > make_move_iterator (_Iterator __i)`
- `template<typename _T1, typename _T2 >`
`constexpr pair< typename __decay_and_strip< _T1 >::__type, typename __decay_and_strip< _T2 >::__type > make_pair (_T1 &&__x, _T2 &&__y)`
- `template<typename _Iterator >`
`reverse_iterator< _Iterator > make_reverse_iterator (_Iterator __i)`
- `template<typename _Tp, typename... _Args>`
`shared_ptr< _Tp > make_shared (_Args &&...__args)`
- `template<typename... _Elements>`
`constexpr tuple< typename __decay_and_strip< _Elements >::__type... > make_tuple (_Elements &&...__args)`
- `template<typename _Tp, typename... _Args>`
`_MakeUniq< _Tp >::__single_object make_unique (_Args &&...__args)`
- `template<typename _Tp >`
`_MakeUniq< _Tp >::__array make_unique (size_t __num)`
- `template<typename _Tp, typename... _Args>`
`_MakeUniq< _Tp >::__invalid_type make_unique (_Args &&...)=delete`
- `template<typename _Tp >`
`_GLIBCXX14_CONSTEXPR const _Tp & max (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp, typename _Compare >`
`_GLIBCXX14_CONSTEXPR const _Tp & max (const _Tp &__a, const _Tp &__b, _Compare __comp)`
- `template<typename _Tp >`
`_GLIBCXX14_CONSTEXPR _Tp max (initializer_list< _Tp >)`
- `template<typename _Tp, typename _Compare >`
`_GLIBCXX14_CONSTEXPR _Tp max (initializer_list< _Tp >, _Compare)`
- `template<typename _Filter >`
`_GLIBCXX14_CONSTEXPR _Filter max_element (_Filter, _Filter)`
- `template<typename _Filter, typename _Compare >`
`_GLIBCXX14_CONSTEXPR _Filter max_element (_Filter, _Filter, _Compare)`
- `template<typename _ForwardIterator >`
`_GLIBCXX14_CONSTEXPR _ForwardIterator max_element (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`_GLIBCXX14_CONSTEXPR _ForwardIterator max_element (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _Tp, typename _Class >`
`_Mem_fn< _Tp _Class::* > mem_fn (_Tp _Class::*__pm) noexcept`
- `template<typename _Ret, typename _Tp >`
`mem_fun_t< _Ret, _Tp > mem_fun (_Ret(_Tp::*__f)())`
- `template<typename _Ret, typename _Tp >`
`const_mem_fun_t< _Ret, _Tp > mem_fun (_Ret(_Tp::*__f)() const)`
- `template<typename _Ret, typename _Tp, typename _Arg >`
`mem_fun1_t< _Ret, _Tp, _Arg > mem_fun (_Ret(_Tp::*__f)(_Arg))`
- `template<typename _Ret, typename _Tp, typename _Arg >`
`const_mem_fun1_t< _Ret, _Tp, _Arg > mem_fun (_Ret(_Tp::*__f)(_Arg) const)`
- `template<typename _Ret, typename _Tp >`
`mem_fun_ref_t< _Ret, _Tp > mem_fun_ref (_Ret(_Tp::*__f)())`
- `template<typename _Ret, typename _Tp >`
`const_mem_fun_ref_t< _Ret, _Tp > mem_fun_ref (_Ret(_Tp::*__f)() const)`

- `template<typename _Ret, typename _Tp, typename _Arg >`
`mem_fun1_ref_t< _Ret, _Tp, _Arg > mem_fun_ref (_Ret(_Tp::*__f)(__Arg))`
- `template<typename _Ret, typename _Tp, typename _Arg >`
`const_mem_fun1_ref_t< _Ret, _Tp, _Arg > mem_fun_ref (_Ret(_Tp::*__f)(__Arg) const)`
- `void * memchr (void * __s, int __c, size_t __n)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter merge (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`_OIter merge (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`_OutputIterator merge (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator merge (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _Tp >`
`_GLIBCXX14_CONSTEXPR const _Tp & min (const _Tp & __a, const _Tp & __b)`
- `template<typename _Tp, typename _Compare >`
`_GLIBCXX14_CONSTEXPR const _Tp & min (const _Tp & __a, const _Tp & __b, _Compare __comp)`
- `template<typename _Tp >`
`_GLIBCXX14_CONSTEXPR _Tp min (initializer_list< _Tp >)`
- `template<typename _Tp, typename _Compare >`
`_GLIBCXX14_CONSTEXPR _Tp min (initializer_list< _Tp >, _Compare)`
- `template<typename _Filter >`
`_GLIBCXX14_CONSTEXPR _Filter min_element (_Filter, _Filter)`
- `template<typename _Filter, typename _Compare >`
`_GLIBCXX14_CONSTEXPR _Filter min_element (_Filter, _Filter, _Compare)`
- `template<typename _ForwardIterator >`
`_GLIBCXX14_CONSTEXPR _ForwardIterator min_element (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`_GLIBCXX14_CONSTEXPR _ForwardIterator min_element (_ForwardIterator __first, _ForwardIterator __last, ←
_Compare __comp)`
- `template<typename _Tp >`
`_GLIBCXX14_CONSTEXPR pair< const _Tp &, const _Tp & > minmax (const _Tp & __a, const _Tp & __b)`
- `template<typename _Tp, typename _Compare >`
`_GLIBCXX14_CONSTEXPR pair< const _Tp &, const _Tp & > minmax (const _Tp & __a, const _Tp & __b, ←
_Compare __comp)`
- `template<typename _Tp >`
`_GLIBCXX14_CONSTEXPR pair< _Tp, _Tp > minmax (initializer_list< _Tp >)`
- `template<typename _Tp, typename _Compare >`
`_GLIBCXX14_CONSTEXPR pair< _Tp, _Tp > minmax (initializer_list< _Tp >, _Compare)`
- `template<typename _Filter >`
`_GLIBCXX14_CONSTEXPR pair< _Filter, _Filter > minmax_element (_Filter, _Filter)`
- `template<typename _Filter, typename _Compare >`
`_GLIBCXX14_CONSTEXPR pair< _Filter, _Filter > minmax_element (_Filter, _Filter, _Compare)`
- `template<typename _ForwardIterator >`
`_GLIBCXX14_CONSTEXPR pair< _ForwardIterator, _ForwardIterator > minmax_element (_ForwardIterator ←
__first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`_GLIBCXX14_CONSTEXPR pair< _ForwardIterator, _ForwardIterator > minmax_element (_ForwardIterator ←
__first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _Iter1, typename _Iter2 >`
`pair< _Iter1, _Iter2 > mismatch (_Iter1, _Iter1, _Iter2)`

- `template<typename _Iter1, typename _Iter2, typename _BinaryPredicate >`
`pair< _Iter1, _Iter2 > mismatch (_Iter1, _Iter1, _Iter2, _BinaryPredicate)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`pair< _InputIterator1, _InputIterator2 > mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`pair< _InputIterator1, _InputIterator2 > mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`pair< _InputIterator1, _InputIterator2 > mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`pair< _InputIterator1, _InputIterator2 > mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _BinaryPredicate __binary_pred)`
- `float modf (float __x, float * __iptr)`
- `long double modf (long double __x, long double * __iptr)`
- `template<typename _Tp >`
`constexpr std::remove_reference< _Tp >::type && move (_Tp && __t) noexcept`
- `template<typename _Tp >`
`_Deque_iterator< _Tp, _Tp &, _Tp * > move (_Deque_iterator< _Tp, _Tp &, _Tp * > __first, _Deque_iterator< _Tp, _Tp &, _Tp * > __last, _Deque_iterator< _Tp, _Tp &, _Tp * > __result)`
- `template<typename _II, typename _OI >`
`_OI move (_II __first, _II __last, _OI __result)`
- `template<typename _Tp >`
`_Deque_iterator< _Tp, _Tp &, _Tp * > move (_Deque_iterator< _Tp, const _Tp &, const _Tp * > __first, _Deque_iterator< _Tp, const _Tp &, const _Tp * > __last, _Deque_iterator< _Tp, _Tp &, _Tp * > __result)`
- `template<typename _Tp >`
`_Deque_iterator< _Tp, _Tp &, _Tp * > move_backward (_Deque_iterator< _Tp, _Tp &, _Tp * > __first, _Deque_iterator< _Tp, _Tp &, _Tp * > __last, _Deque_iterator< _Tp, _Tp &, _Tp * > __result)`
- `template<typename _BI1, typename _BI2 >`
`_BI2 move_backward (_BI1 __first, _BI1 __last, _BI2 __result)`
- `template<typename _Tp >`
`_Deque_iterator< _Tp, _Tp &, _Tp * > move_backward (_Deque_iterator< _Tp, const _Tp &, const _Tp * > __first, _Deque_iterator< _Tp, const _Tp &, const _Tp * > __last, _Deque_iterator< _Tp, _Tp &, _Tp * > __result)`
- `template<typename _Tp >`
`constexpr conditional< __move_if_noexcept_cond< _Tp >::value, const _Tp &, _Tp && >::type move_if_noexcept (_Tp & __x) noexcept`
- `template<typename _ForwardIterator >`
`_ForwardIterator next (_ForwardIterator __x, typename iterator_traits< _ForwardIterator >::difference_type __n=1)`
- `template<typename _BIter >`
`bool next_permutation (_BIter, _BIter)`
- `template<typename _BIter, typename _Compare >`
`bool next_permutation (_BIter, _BIter, _Compare)`
- `template<typename _BidirectionalIterator >`
`bool next_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`bool next_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare __comp)`
- `ios_base & noboolalpha (ios_base & __base)`
- `template<typename _Iter, typename _Predicate >`
`bool none_of (_Iter, _Iter, _Predicate)`

- `template<typename _InputIterator, typename _Predicate >`
`bool none_of (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _Tp >`
`_Tp norm (const complex< _Tp > &)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type norm (_Tp __x)`
- `ios_base & noshowbase (ios_base & __base)`
- `ios_base & noshowpoint (ios_base & __base)`
- `ios_base & noshowpos (ios_base & __base)`
- `ios_base & noskipws (ios_base & __base)`
- `template<typename _Predicate >`
`__GLIBCXX14_CONSTEXPR unary_negate< _Predicate > not1 (const _Predicate & __pred)`
- `template<typename _Predicate >`
`__GLIBCXX14_CONSTEXPR binary_negate< _Predicate > not2 (const _Predicate & __pred)`
- `void notify_all_at_thread_exit (condition_variable &, unique_lock< mutex >)`
- `ios_base & nouitbuf (ios_base & __base)`
- `ios_base & nouppercase (ios_base & __base)`
- `template<typename _RAIter >`
`void nth_element (_RAIter, _RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`void nth_element (_RAIter, _RAIter, _RAIter, _Compare)`
- `template<typename _RandomAccessIterator >`
`void nth_element (_RandomAccessIterator __first, _RandomAccessIterator __nth, _RandomAccessIterator __↵
last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void nth_element (_RandomAccessIterator __first, _RandomAccessIterator __nth, _RandomAccessIterator __↵
last, _Compare __comp)`
- `ios_base & oct (ios_base & __base)`
- `template<class _Tp, class _CharT, class _Traits, class _Dist >`
`bool operator!= (const istream_iterator< _Tp, _CharT, _Traits, _Dist > & __x, const istream_iterator< _Tp, _↵
CharT, _Traits, _Dist > & __y)`
- `template<typename _T1, typename _T2 >`
`bool operator!= (const allocator< _T1 > &, const allocator< _T2 > &) noexcept`
- `template<typename _Tp >`
`bool operator!= (const allocator< _Tp > &, const allocator< _Tp > &) noexcept`
- `template<typename _CharT, typename _Traits >`
`bool operator!= (const istreambuf_iterator< _CharT, _Traits > & __a, const istreambuf_iterator< _CharT, _Traits
> & __b)`
- `template<typename _StateT >`
`bool operator!= (const fpos< _StateT > & __lhs, const fpos< _StateT > & __rhs)`
- `bool operator!= (thread::id __x, thread::id __y) noexcept`
- `template<typename _Tp, std::size_t _Nm >`
`bool operator!= (const array< _Tp, _Nm > & __one, const array< _Tp, _Nm > & __two)`
- `template<typename _Tp >`
`bool operator!= (const _Fwd_list_iterator< _Tp > & __x, const _Fwd_list_const_iterator< _Tp > & __y) noexcept`
- `template<typename _Tp, typename _Ref, typename _Ptr >`
`bool operator!= (const _Deque_iterator< _Tp, _Ref, _Ptr > & __x, const _Deque_iterator< _Tp, _Ref, _Ptr >
& __y) noexcept`
- `template<typename _Tp, typename _RefL, typename _PtrL, typename _RefR, typename _PtrR >`
`bool operator!= (const _Deque_iterator< _Tp, _RefL, _PtrL > & __x, const _Deque_iterator< _Tp, _RefR, _PtrR
> & __y) noexcept`
- `template<typename _Val >`
`bool operator!= (const _List_iterator< _Val > & __x, const _List_const_iterator< _Val > & __y) noexcept`

- `template<typename _Tp, typename _Seq >`
`bool operator!= (const stack< _Tp, _Seq > &__x, const stack< _Tp, _Seq > &__y)`
- `template<typename _Iterator >`
`bool operator!= (const reverse_iterator< _Iterator > &__x, const reverse_iterator< _Iterator > &__y)`
- `bool operator!= (const error_code &__lhs, const error_code &__rhs) noexcept`
- `bool operator!= (const error_code &__lhs, const error_condition &__rhs) noexcept`
- `bool operator!= (const error_condition &__lhs, const error_code &__rhs) noexcept`
- `template<typename _Tp, typename _Seq >`
`bool operator!= (const queue< _Tp, _Seq > &__x, const queue< _Tp, _Seq > &__y)`
- `bool operator!= (const error_condition &__lhs, const error_condition &__rhs) noexcept`
- `template<typename _Val >`
`bool operator!= (const _Rb_tree_iterator< _Val > &__x, const _Rb_tree_const_iterator< _Val > &__y) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator!= (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool operator!= (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _IteratorL, typename _IteratorR >`
`bool operator!= (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _Tp >`
`bool operator!= (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m>`
`bool operator!= (const std::linear_congruential_engine< _UIntType, __a, __c, __m > &__lhs, const std::linear_congruential_engine< _UIntType, __a, __c, __m > &__rhs)`
- `template<class _Dom >`
`_Expr< _BinClos< __not_equal_to, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __not_equal_to, typename _Dom::value_type >::result_type > operator!= (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __not_equal_to, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __not_equal_to, typename _Dom1::value_type >::result_type > operator!= (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __not_equal_to, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< __not_equal_to, typename _Dom::value_type >::result_type > operator!= (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __not_equal_to, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __not_equal_to, typename _Dom::value_type >::result_type > operator!= (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __not_equal_to, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< __not_equal_to, typename _Dom::value_type >::result_type > operator!= (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<typename _T1, typename _T2 >`
`constexpr bool operator!= (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename _OutA1, typename _OutA2, typename... _InA>`
`bool operator!= (const scoped_allocator_adaptor< _OutA1, _InA... > &__a, const scoped_allocator_adaptor< _OutA2, _InA... > &__b) noexcept`
- `template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f>`
`bool operator!= (const std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f > &__lhs, const std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f > &__rhs)`

- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`bool operator!= (const unique_ptr< _Tp, _Dp > &__x, const unique_ptr< _Up, _Ep > &__y)`
- `template<typename _Tp, typename _Dp >`
`bool operator!= (const unique_ptr< _Tp, _Dp > &__x, nullptr_t) noexcept`
- `template<typename _Tp, typename _Dp >`
`bool operator!= (nullptr_t, const unique_ptr< _Tp, _Dp > &__x) noexcept`
- `template<typename _UIntType, size_t __w, size_t __s, size_t __r>`
`bool operator!= (const std::subtract_with_carry_engine< _UIntType, __w, __s, __r > &__lhs, const std::subtract_with_carry_engine< _UIntType, __w, __s, __r > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator!= (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator!= (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Bilter >`
`bool operator!= (const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator!= (const multimap< _Key, _Tp, _Compare, _Alloc > &__x, const multimap< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
`bool operator!= (const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _RandomNumberEngine, size_t __p, size_t __r>`
`bool operator!= (const std::discard_block_engine< _RandomNumberEngine, __p, __r > &__lhs, const std::discard_block_engine< _RandomNumberEngine, __p, __r > &__rhs)`
- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
`bool operator!= (const sub_match< _Bi_iter > &__lhs, const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__rhs)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool operator!= (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _Iterator >`
`bool operator!= (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _Tp >`
`_Expr< _BinClos< __not_equal_to, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __not_equal_to, _Tp >::result_type > operator!= (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __not_equal_to, _ValArray, Constant, _Tp, _Tp >, typename __fun< __not_equal_to, _Tp >::result_type > operator!= (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __not_equal_to, Constant, _ValArray, _Tp, _Tp >, typename __fun< __not_equal_to, _Tp >::result_type > operator!= (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Bi_iter >`
`bool operator!= (typename iterator_traits< _Bi_iter >::value_type const * __lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>`
`bool operator!= (const __shared_ptr< _Tp1, _Lp > &__a, const __shared_ptr< _Tp2, _Lp > &__b) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator!= (const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator!= (nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`
- `template<typename _RandomNumberEngine, size_t __w, typename _UIntType >`
`bool operator!= (const std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > &__lhs, const std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > &__rhs)`

- `template<typename _Bi_iter >`
`bool operator!= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const * __rhs)`
- `template<typename _Key, typename _Val, typename _KeyOfValue, typename _Compare, typename _Alloc >`
`bool operator!= (const Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc > &__x, const Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator!= (const map< _Key, _Tp, _Compare, _Alloc > &__x, const map< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Bi_iter >`
`bool operator!= (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool operator!= (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename _Tp, typename _Alloc >`
`bool operator!= (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Bi_iter >`
`bool operator!= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<class _Value, class _Hash, class _Pred, class _Alloc >`
`bool operator!= (const unordered_set< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_set< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<class _Value, class _Hash, class _Pred, class _Alloc >`
`bool operator!= (const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _RandomNumberEngine, size_t __k>`
`bool operator!= (const std::shuffle_order_engine< _RandomNumberEngine, __k > &__lhs, const std::shuffle_order_engine< _RandomNumberEngine, __k > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator!= (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `template<typename _IntType >`
`bool operator!= (const std::uniform_int_distribution< _IntType > &__d1, const std::uniform_int_distribution< _IntType > &__d2)`
- `template<class _Key, class _Tp, class _Hash, class _Pred, class _Alloc >`
`bool operator!= (const unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, const unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > &__y)`
- `template<class _Key, class _Tp, class _Hash, class _Pred, class _Alloc >`
`bool operator!= (const unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, const unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _IntType >`
`bool operator!= (const std::uniform_real_distribution< _IntType > &__d1, const std::uniform_real_distribution< _IntType > &__d2)`
- `template<typename _Tp, typename _Alloc >`
`bool operator!= (const list< _Tp, _Alloc > &__x, const list< _Tp, _Alloc > &__y)`
- `template<typename _Bi_iter, class _Alloc >`
`bool operator!= (const match_results< _Bi_iter, _Alloc > &__m1, const match_results< _Bi_iter, _Alloc > &__m2)`
- `template<typename _RealType >`
`bool operator!= (const std::normal_distribution< _RealType > &__d1, const std::normal_distribution< _RealType > &__d2)`
- `template<typename _Res, typename... _Args>`
`bool operator!= (const function< _Res(_Args...) > &__f, nullptr_t) noexcept`

- `template<typename _Res, typename... _Args>`
`bool operator!= (nullptr_t, const function< _Res(_Args...)> &__f) noexcept`
- `template<typename _Tp, typename _Alloc >`
`bool operator!= (const deque< _Tp, _Alloc > &__x, const deque< _Tp, _Alloc > &__y)`
- `template<typename _RealType >`
`bool operator!= (const std::lognormal_distribution< _RealType > &__d1, const std::lognormal_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool operator!= (const std::gamma_distribution< _RealType > &__d1, const std::gamma_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool operator!= (const std::chi_squared_distribution< _RealType > &__d1, const std::chi_squared_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool operator!= (const std::cauchy_distribution< _RealType > &__d1, const std::cauchy_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool operator!= (const std::fisher_f_distribution< _RealType > &__d1, const std::fisher_f_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool operator!= (const std::student_t_distribution< _RealType > &__d1, const std::student_t_distribution< _RealType > &__d2)`
- `bool operator!= (const std::bernoulli_distribution &__d1, const std::bernoulli_distribution &__d2)`
- `template<typename _IntType >`
`bool operator!= (const std::binomial_distribution< _IntType > &__d1, const std::binomial_distribution< _IntType > &__d2)`
- `template<typename _IntType >`
`bool operator!= (const std::geometric_distribution< _IntType > &__d1, const std::geometric_distribution< _IntType > &__d2)`
- `template<typename _IntType >`
`bool operator!= (const std::negative_binomial_distribution< _IntType > &__d1, const std::negative_binomial_distribution< _IntType > &__d2)`
- `template<typename _IntType >`
`bool operator!= (const std::poisson_distribution< _IntType > &__d1, const std::poisson_distribution< _IntType > &__d2)`
- `template<typename _RealType >`
`bool operator!= (const std::exponential_distribution< _RealType > &__d1, const std::exponential_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool operator!= (const std::weibull_distribution< _RealType > &__d1, const std::weibull_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool operator!= (const std::extreme_value_distribution< _RealType > &__d1, const std::extreme_value_distribution< _RealType > &__d2)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator!= (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator!= (const _CharT *__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator!= (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const _CharT *__rhs)`
- `template<typename _IntType >`
`bool operator!= (const std::discrete_distribution< _IntType > &__d1, const std::discrete_distribution< _IntType > &__d2)`

- `template<typename _RealType >`
`bool operator!= (const std::piecewise_constant_distribution< _RealType > &__d1, const std::piecewise_constant_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool operator!= (const std::piecewise_linear_distribution< _RealType > &__d1, const std::piecewise_linear_distribution< _RealType > &__d2)`
- `template<class _Dom >`
`_Expr< _BinClos< __modulus, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __modulus, typename _Dom::value_type >::result_type > operator% (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __modulus, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< __modulus, typename _Dom::value_type >::result_type > operator% (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __modulus, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __modulus, typename _Dom1::value_type >::result_type > operator% (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __modulus, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __modulus, typename _Dom::value_type >::result_type > operator% (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< __modulus, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< __modulus, typename _Dom::value_type >::result_type > operator% (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __modulus, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __modulus, _Tp >::result_type > operator% (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __modulus, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __modulus, _Tp >::result_type > operator% (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __modulus, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __modulus, _Tp >::result_type > operator% (const _Tp &__t, const valarray< _Tp > &__v)`
- `constexpr memory_order operator& (memory_order __m, __memory_order_modifier __mod)`
- `constexpr _ios_Fmtflags operator& (_ios_Fmtflags __a, _ios_Fmtflags __b)`
- `constexpr _ios_Openmode operator& (_ios_Openmode __a, _ios_Openmode __b)`
- `constexpr launch operator& (launch __x, launch __y)`
- `constexpr _ios_istate operator& (_ios_istate __a, _ios_istate __b)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __bitwise_and, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __bitwise_and, typename _Dom1::value_type >::result_type > operator& (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __bitwise_and, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< __bitwise_and, typename _Dom::value_type >::result_type > operator& (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __bitwise_and, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __bitwise_and, typename _Dom::value_type >::result_type > operator& (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`

- `template<class _Dom >`
`_Expr< _BinClos< __bitwise_and, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__bitwise_and, typename _Dom::value_type >::result_type > operator& (const valarray< typename _Dom::`
`value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< __bitwise_and, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun<`
`__bitwise_and, typename _Dom::value_type >::result_type > operator& (const _Expr< _Dom, typename`
`_Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __bitwise_and, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __bitwise_and, _Tp`
`>::result_type > operator& (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __bitwise_and, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __bitwise_and, _Tp`
`>::result_type > operator& (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __bitwise_and, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __bitwise_and, _Tp`
`>::result_type > operator& (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __logical_and, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __logical_and, typename`
`_Dom1::value_type >::result_type > operator&& (const _Expr< _Dom1, typename _Dom1::value_type > &__v,`
`const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __logical_and, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__logical_and, typename _Dom::value_type >::result_type > operator&& (const valarray< typename _Dom::`
`value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< __logical_and, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__logical_and, typename _Dom::value_type >::result_type > operator&& (const typename _Dom::value_type`
`&__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __logical_and, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun<`
`__logical_and, typename _Dom::value_type >::result_type > operator&& (const _Expr< _Dom, typename _`
`Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __logical_and, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun<`
`__logical_and, typename _Dom::value_type >::result_type > operator&& (const _Expr< _Dom, typename _`
`Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __logical_and, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __logical_and, _Tp >`
`::result_type > operator&& (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __logical_and, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __logical_and, _Tp`
`>::result_type > operator&& (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __logical_and, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __logical_and, _Tp`
`>::result_type > operator&& (const valarray< _Tp > &__v, const _Tp &__t)`
- `const _ios_Fmtflags & operator&= (_ios_Fmtflags &__a, _ios_Fmtflags __b)`
- `const _ios_Openmode & operator&= (_ios_Openmode &__a, _ios_Openmode __b)`
- `launch & operator&= (launch &__x, launch __y)`
- `const _ios_ostate & operator&= (_ios_ostate &__a, _ios_ostate __b)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __multiplies, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __multiplies, typename`
`_Dom1::value_type >::result_type > operator* (const _Expr< _Dom1, typename _Dom1::value_type > &__v,`
`const _Expr< _Dom2, typename _Dom2::value_type > &__w)`

- `template<class _Dom >`
`_Expr< _BinClos< __multiplies, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun<`
`__multiplies, typename _Dom::value_type >::result_type > operator* (const _Expr< _Dom, typename _Dom::`
`value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __multiplies, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__multiplies, typename _Dom::value_type >::result_type > operator* (const typename _Dom::value_type &__t,`
`const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __multiplies, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __`
`multiplies, typename _Dom::value_type >::result_type > operator* (const valarray< typename _Dom::value`
`type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< __multiplies, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun<`
`__multiplies, typename _Dom::value_type >::result_type > operator* (const _Expr< _Dom, typename _Dom::`
`value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __multiplies, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __multiplies, _Tp >::`
`result_type > operator* (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __multiplies, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __multiplies, _Tp >::`
`result_type > operator* (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __multiplies, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __multiplies, _Tp >::`
`result_type > operator* (const valarray< _Tp > &__v, const _Tp &__t)`
- `_Bit_iterator operator+ (ptrdiff_t __n, const _Bit_iterator &__x)`
- `template<typename _Iterator >`
`reverse_iterator< _Iterator > operator+ (typename reverse_iterator< _Iterator >::difference_type __n, const`
`reverse_iterator< _Iterator > &__x)`
- `template<typename _Tp, typename _Ref, typename _Ptr >`
`Deque_iterator< _Tp, _Ref, _Ptr > operator+ (ptrdiff_t __n, const Deque_iterator< _Tp, _Ref, _Ptr > &__x)`
`noexcept`
- `_Bit_const_iterator operator+ (ptrdiff_t __n, const _Bit_const_iterator &__x)`
- `template<class _Dom >`
`_Expr< _BinClos< __plus, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< __plus,`
`typename _Dom::value_type >::result_type > operator+ (const _Expr< _Dom, typename _Dom::value_type >`
`&__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __plus, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __plus,`
`typename _Dom::value_type >::result_type > operator+ (const valarray< typename _Dom::value_type > &__v,`
`const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __plus, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __plus, typename _Dom1::`
`value_type >::result_type > operator+ (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const`
`_Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __plus, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< __plus,`
`typename _Dom::value_type >::result_type > operator+ (const _Expr< _Dom, typename _Dom::value_type >`
`&__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __plus, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __`
`plus, typename _Dom::value_type >::result_type > operator+ (const typename _Dom::value_type &__t, const`
`_Expr< _Dom, typename _Dom::value_type > &__v)`

- `template<typename _Tp >`
`complex< _Tp > operator+ (const complex< _Tp > &__x)`
- `template<typename _Tp >`
`_Expr< _BinClos< __plus, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __plus, _Tp >::result_type >`
`operator+ (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __plus, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __plus, _Tp >::result_type >`
`operator+ (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __plus, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __plus, _Tp >::result_type >`
`operator+ (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Iterator >`
`move_iterator< _Iterator > operator+ (typename move_iterator< _Iterator >::difference_type __n, const move_iterator< _Iterator > &__x)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > operator+ (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > operator+ (const _CharT *__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > operator+ (_CharT __lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > operator+ (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > operator+ (const basic_string< _CharT, _Traits, _Alloc > &__lhs, _CharT __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > operator+ (basic_string< _CharT, _Traits, _Alloc > &&__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > operator+ (const basic_string< _CharT, _Traits, _Alloc > &__lhs, basic_string< _CharT, _Traits, _Alloc > &&__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > operator+ (basic_string< _CharT, _Traits, _Alloc > &&__lhs, basic_string< _CharT, _Traits, _Alloc > &&__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > operator+ (const _CharT *__lhs, basic_string< _CharT, _Traits, _Alloc > &&__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > operator+ (_CharT __lhs, basic_string< _CharT, _Traits, _Alloc > &&__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > operator+ (basic_string< _CharT, _Traits, _Alloc > &&__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > operator+ (basic_string< _CharT, _Traits, _Alloc > &&__lhs, _CharT __rhs)`
- `ptrdiff_t operator- (const _Bit_iterator_base &__x, const _Bit_iterator_base &__y)`
- `template<typename _Iterator >`
`auto operator- (const reverse_iterator< _Iterator > &__x, const reverse_iterator< _Iterator > &__y) ->`
`decltype(__x.base()-__y.base())`

- `template<typename _Tp, typename _Ref, typename _Ptr >`
`_Deque_iterator< _Tp, _Ref, _Ptr >::difference_type operator- (const _Deque_iterator< _Tp, _Ref, _Ptr > &__x, const _Deque_iterator< _Tp, _Ref, _Ptr > &__y) noexcept`
- `template<typename _Tp, typename _RefL, typename _PtrL, typename _RefR, typename _PtrR >`
`_Deque_iterator< _Tp, _RefL, _PtrL >::difference_type operator- (const _Deque_iterator< _Tp, _RefL, _PtrL > &__x, const _Deque_iterator< _Tp, _RefR, _PtrR > &__y) noexcept`
- `template<typename _IteratorL, typename _IteratorR >`
`auto operator- (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y) -> decltype(__y.base()-__x.base())`
- `template<class _Dom >`
`_Expr< _BinClos< __minus, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __minus, typename _Dom::value_type >::result_type > operator- (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __minus, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< __minus, typename _Dom::value_type >::result_type > operator- (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __minus, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __minus, typename _Dom1::value_type >::result_type > operator- (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __minus, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< __minus, typename _Dom::value_type >::result_type > operator- (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __minus, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __minus, typename _Dom::value_type >::result_type > operator- (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`complex< _Tp > operator- (const complex< _Tp > &__x)`
- `template<typename _Tp >`
`_Expr< _BinClos< __minus, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __minus, _Tp >::result_type > operator- (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __minus, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __minus, _Tp >::result_type > operator- (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __minus, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __minus, _Tp >::result_type > operator- (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _IteratorL, typename _IteratorR >`
`auto operator- (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y) -> decltype(__x.base()-__y.base())`
- `template<typename _Iterator >`
`auto operator- (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y) -> decltype(__x.base()-__y.base())`
- `template<class _Dom >`
`_Expr< _BinClos< __divides, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __divides, typename _Dom::value_type >::result_type > operator/ (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< __divides, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< __divides, typename _Dom::value_type >::result_type > operator/ (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`

- `template<class _Dom >`
`_Expr< _BinClos< __divides, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< __divides, typename _Dom::value_type >::result_type > operator/ (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __divides, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __divides, typename _Dom::value_type >::result_type > operator/ (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __divides, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __divides, typename _Dom1::value_type >::result_type > operator/ (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __divides, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __divides, _Tp >::result_type > operator/ (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __divides, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __divides, _Tp >::result_type > operator/ (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __divides, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __divides, _Tp >::result_type > operator/ (const _Tp &__t, const valarray< _Tp > &__v)`
- `bool operator< (const error_code &__lhs, const error_code &__rhs) noexcept`
- `template<typename _Tp, std::size_t _Nm >`
`bool operator< (const array< _Tp, _Nm > &__a, const array< _Tp, _Nm > &__b)`
- `bool operator< (const error_condition &__lhs, const error_condition &__rhs) noexcept`
- `template<typename _Tp, typename _Ref, typename _Ptr >`
`bool operator< (const Deque_iterator< _Tp, _Ref, _Ptr > &__x, const Deque_iterator< _Tp, _Ref, _Ptr > &__y) noexcept`
- `template<typename _Tp, typename _Seq >`
`bool operator< (const stack< _Tp, _Seq > &__x, const stack< _Tp, _Seq > &__y)`
- `template<typename _Iterator >`
`bool operator< (const reverse_iterator< _Iterator > &__x, const reverse_iterator< _Iterator > &__y)`
- `template<typename _Tp, typename _RefL, typename _PtrL, typename _RefR, typename _PtrR >`
`bool operator< (const Deque_iterator< _Tp, _RefL, _PtrL > &__x, const Deque_iterator< _Tp, _RefR, _PtrR > &__y) noexcept`
- `template<typename _Tp, typename _Seq >`
`bool operator< (const queue< _Tp, _Seq > &__x, const queue< _Tp, _Seq > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool operator< (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator< (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool operator< (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool operator< (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<class _Dom >`
`_Expr< _BinClos< __less, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __less, typename _Dom::value_type >::result_type > operator< (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __less, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< __less, typename _Dom::value_type >::result_type > operator< (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`

- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __less, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __less, typename _Dom1::value_type >::result_type > operator< (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __less, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< __less, typename _Dom::value_type >::result_type > operator< (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __less, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __less, typename _Dom::value_type >::result_type > operator< (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _T1, typename _T2 >`
`constexpr bool operator< (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`bool operator< (const unique_ptr< _Tp, _Dp > &__x, const unique_ptr< _Up, _Ep > &__y)`
- `template<typename _Tp, typename _Dp >`
`bool operator< (const unique_ptr< _Tp, _Dp > &__x, nullptr_t)`
- `template<typename _Tp, typename _Dp >`
`bool operator< (nullptr_t, const unique_ptr< _Tp, _Dp > &__x)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator< (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator< (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Bilter >`
`bool operator< (const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator< (const multimap< _Key, _Tp, _Compare, _Alloc > &__x, const multimap< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
`bool operator< (const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter, class _Ch_traits, class _Ch_alloc >`
`bool operator< (const sub_match< _Bi_iter > &__lhs, const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__rhs)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool operator< (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _Iterator >`
`bool operator< (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _Tp >`
`_Expr< _BinClos< __less, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __less, _Tp >::result_type > operator< (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __less, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __less, _Tp >::result_type > operator< (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __less, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __less, _Tp >::result_type > operator< (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Bi_iter >`
`bool operator< (typename iterator_traits< _Bi_iter >::value_type const * __lhs, const sub_match< _Bi_iter > &__rhs)`

- `template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>`
`bool operator< (const __shared_ptr< _Tp1, _Lp > &__a, const __shared_ptr< _Tp2, _Lp > &__b) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator< (const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator< (nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`
- `template<typename _Bi_iter >`
`bool operator< (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const * __rhs)`
- `template<typename _Key, typename _Val, typename _KeyOfValue, typename _Compare, typename _Alloc >`
`bool operator< (const Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc > &__x, const Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator< (const map< _Key, _Tp, _Compare, _Alloc > &__x, const map< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Bi_iter >`
`bool operator< (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool operator< (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename _Tp, typename _Alloc >`
`bool operator< (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Bi_iter >`
`bool operator< (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator< (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool operator< (const list< _Tp, _Alloc > &__x, const list< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool operator< (const deque< _Tp, _Alloc > &__x, const deque< _Tp, _Alloc > &__y)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator< (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator< (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const _CharT * __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator< (const _CharT * __lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, _Resetiosflags __f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, _Setiosflags __f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, _Setbase __f)`
- `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const linear_congruential_engine< _UIntType, __a, __c, __m > &__lcr)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, _Setfill< _CharT > __f)`

- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, const error_code`
`&__e)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, _Setprecision`
`__f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, _Setw __f)`
- `template<class _CharT, class _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__out, thread::id __id)`
- `template<typename _CharT, typename _Traits, typename _MoneyT >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, _Put_money<`
`_MoneyT > __f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, _Put_time< _↵`
`CharT > __f)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __shift_left, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __shift_left, typename _↵`
`_Dom1::value_type >::result_type > operator<< (const _Expr< _Dom1, typename _Dom1::value_type > &__v,`
`const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __shift_left, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__shift_left, typename _Dom::value_type >::result_type > operator<< (const typename _Dom::value_type &↵`
`__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __shift_left, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun<`
`__shift_left, typename _Dom::value_type >::result_type > operator<< (const _Expr< _Dom, typename _↵`
`_Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __shift_left, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< _↵`
`__shift_left, typename _Dom::value_type >::result_type > operator<< (const _Expr< _Dom, typename _Dom↵`
`::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __shift_left, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< _↵`
`__shift_left, typename _Dom::value_type >::result_type > operator<< (const valarray< typename _Dom::value↵`
`_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _U↵`
`IntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f > &__x)`
- `template<typename _Ch, typename _Tr, typename _Tp, _Lock_policy _Lp>`
`std::basic_ostream< _Ch, _Tr > & operator<< (std::basic_ostream< _Ch, _Tr > &__os, const __shared_ptr<`
`_Tp, _Lp > &__p)`
- `template<typename _Tp, typename _CharT, class _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, const complex<`
`_Tp > &__x)`
- `template<typename _UIntType, size_t __w, size_t __s, size_t __r, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`subtract_with_carry_engine< _UIntType, __w, __s, __r > &__x)`
- `template<typename _CharT, typename _Traits, typename _Tp >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &&__os, const _Tp &__x)`
- `template<typename _RandomNumberEngine, size_t __p, size_t __r, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`discard_block_engine< _RandomNumberEngine, __p, __r > &__x)`

- `template<typename _RandomNumberEngine, size_t __k, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`shuffle_order_engine< _RandomNumberEngine, __k > &__x)`
- `template<typename _Tp >`
`_Expr< _BinClos< __shift_left, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __shift_left, _Tp >::result<`
`_type > operator<< (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __shift_left, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __shift_left, _Tp >::result<`
`_type > operator<< (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __shift_left, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __shift_left, _Tp >::result<`
`_type > operator<< (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`negative_binomial_distribution< _IntType > &__x)`
- `template<typename _RandomNumberEngine, size_t __w, typename _UIntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`poisson_distribution< _IntType > &__x)`
- `template<typename _Ch_type, typename _Ch_traits, typename _Bi_iter >`
`basic_ostream< _Ch_type, _Ch_traits > & operator<< (basic_ostream< _Ch_type, _Ch_traits > &__os, const`
`sub_match< _Bi_iter > &__m)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`binomial_distribution< _IntType > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &, const std<`
`::uniform_int_distribution< _IntType > &)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &, const std<`
`::uniform_real_distribution< _RealType > &)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`normal_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`lognormal_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`chi_squared_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`fisher_f_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`student_t_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`gamma_distribution< _RealType > &__x)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, const __gnu<`
`cxx::__versa_string< _CharT, _Traits, _Alloc, _Base > &__str)`

- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`discrete_distribution< _IntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`piecewise_constant_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::cauchy_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`piecewise_linear_distribution< _RealType > &__x)`
- `template<typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::bernoulli_distribution &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::geometric_distribution< _IntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::exponential_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::weibull_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::extreme_value_distribution< _RealType > &__x)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__os, const basic_↵`
`string< _CharT, _Traits, _Alloc > &__str)`
- `bool operator<= (thread::id __x, thread::id __y) noexcept`
- `template<typename _Tp, std::size_t _Nm>`
`bool operator<= (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _Tp, typename _Seq >`
`bool operator<= (const stack< _Tp, _Seq > &__x, const stack< _Tp, _Seq > &__y)`
- `template<typename _Iterator >`
`bool operator<= (const reverse_iterator< _Iterator > &__x, const reverse_iterator< _Iterator > &__y)`
- `template<typename _Tp, typename _Ref, typename _Ptr >`
`bool operator<= (const _Deque_iterator< _Tp, _Ref, _Ptr > &__x, const _Deque_iterator< _Tp, _Ref, _Ptr >`
`&__y) noexcept`
- `template<typename _Tp, typename _RefL, typename _PtrL, typename _RefR, typename _PtrR >`
`bool operator<= (const _Deque_iterator< _Tp, _RefL, _PtrL > &__x, const _Deque_iterator< _Tp, _RefR, _↵`
`PtrR > &__y) noexcept`
- `template<typename _Tp, typename _Seq >`
`bool operator<= (const queue< _Tp, _Seq > &__x, const queue< _Tp, _Seq > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool operator<= (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator<= (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool operator<= (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool operator<= (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`

- `template<class _Dom >`
`_Expr< _BinClos< __less_equal, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun<`
`__less_equal, typename _Dom::value_type >::result_type > operator<= (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __less_equal, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__less_equal, typename _Dom::value_type >::result_type > operator<= (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __less_equal, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __less_equal, typename`
`_Dom1::value_type >::result_type > operator<= (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __less_equal, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun<`
`__less_equal, typename _Dom::value_type >::result_type > operator<= (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __less_equal, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__less_equal, typename _Dom::value_type >::result_type > operator<= (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<typename _T1, typename _T2 >`
`constexpr bool operator<= (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`bool operator<= (const unique_ptr< _Tp, _Dp > &__x, const unique_ptr< _Up, _Ep > &__y)`
- `template<typename _Tp, typename _Dp >`
`bool operator<= (const unique_ptr< _Tp, _Dp > &__x, nullptr_t)`
- `template<typename _Tp, typename _Dp >`
`bool operator<= (nullptr_t, const unique_ptr< _Tp, _Dp > &__x)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator<= (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator<= (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Bilter >`
`bool operator<= (const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator<= (const multimap< _Key, _Tp, _Compare, _Alloc > &__x, const multimap< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
`bool operator<= (const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool operator<= (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _Iterator >`
`bool operator<= (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _Bi_iter, class _Ch_traits, class _Ch_alloc >`
`bool operator<= (const sub_match< _Bi_iter > &__lhs, const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__rhs)`
- `template<typename _Tp >`
`_Expr< _BinClos< __less_equal, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __less_equal, _Tp >::result_type > operator<= (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __less_equal, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __less_equal, _Tp >::result_type > operator<= (const _Tp &__t, const valarray< _Tp > &__v)`

- `template<typename _Tp >`
`_Expr< _BinClos< __less_equal, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __less_equal, _Tp >::result_type > operator<= (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Bi_iter >`
`bool operator<= (typename iterator_traits< _Bi_iter >::value_type const * __lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>`
`bool operator<= (const __shared_ptr< _Tp1, _Lp > &__a, const __shared_ptr< _Tp2, _Lp > &__b) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator<= (const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator<= (nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`
- `template<typename _Bi_iter >`
`bool operator<= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const * __rhs)`
- `template<typename _Key, typename _Val, typename _KeyOfValue, typename _Compare, typename _Alloc >`
`bool operator<= (const _Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc > &__x, const _Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator<= (const map< _Key, _Tp, _Compare, _Alloc > &__x, const map< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool operator<= (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename _Bi_iter >`
`bool operator<= (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator<= (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Bi_iter >`
`bool operator<= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator<= (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool operator<= (const list< _Tp, _Alloc > &__x, const list< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool operator<= (const deque< _Tp, _Alloc > &__x, const deque< _Tp, _Alloc > &__y)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator<= (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator<= (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const _CharT * __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator<= (const _CharT * __lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _Tp, typename _CharT, typename _Traits, typename _Dist >`
`bool operator== (const istream_iterator< _Tp, _CharT, _Traits, _Dist > &__x, const istream_iterator< _Tp, _CharT, _Traits, _Dist > &__y)`
- `template<typename _T1, typename _T2 >`
`bool operator== (const allocator< _T1 > &, const allocator< _T2 > &) noexcept`
- `template<typename _Tp >`
`bool operator== (const allocator< _Tp > &, const allocator< _Tp > &) noexcept`

- `template<typename _CharT, typename _Traits >`
`bool operator== (const istreambuf_iterator< _CharT, _Traits > &__a, const istreambuf_iterator< _CharT, _Traits > &__b)`
- `template<typename _StateT >`
`bool operator== (const fpos< _StateT > &__lhs, const fpos< _StateT > &__rhs)`
- `template<typename _Tp, std::size_t _Nm>`
`bool operator== (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _Tp >`
`bool operator== (const _Fwd_list_iterator< _Tp > &__x, const _Fwd_list_const_iterator< _Tp > &__y) noexcept`
- `template<typename _Tp, typename _Ref, typename _Ptr >`
`bool operator== (const _Deque_iterator< _Tp, _Ref, _Ptr > &__x, const _Deque_iterator< _Tp, _Ref, _Ptr > &__y) noexcept`
- `template<typename _Tp, typename _RefL, typename _PtrL, typename _RefR, typename _PtrR >`
`bool operator== (const _Deque_iterator< _Tp, _RefL, _PtrL > &__x, const _Deque_iterator< _Tp, _RefR, _PtrR > &__y) noexcept`
- `template<typename _Tp, typename _Seq >`
`bool operator== (const stack< _Tp, _Seq > &__x, const stack< _Tp, _Seq > &__y)`
- `bool operator== (const error_code &__lhs, const error_code &__rhs) noexcept`
- `template<typename _Val >`
`bool operator== (const _List_iterator< _Val > &__x, const _List_const_iterator< _Val > &__y) noexcept`
- `bool operator== (const error_code &__lhs, const error_condition &__rhs) noexcept`
- `template<typename _Iterator >`
`bool operator== (const reverse_iterator< _Iterator > &__x, const reverse_iterator< _Iterator > &__y)`
- `bool operator== (const error_condition &__lhs, const error_code &__rhs) noexcept`
- `template<typename _Tp, typename _Seq >`
`bool operator== (const queue< _Tp, _Seq > &__x, const queue< _Tp, _Seq > &__y)`
- `bool operator== (const error_condition &__lhs, const error_condition &__rhs) noexcept`
- `template<typename _Val >`
`bool operator== (const _Rb_tree_iterator< _Val > &__x, const _Rb_tree_const_iterator< _Val > &__y) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator== (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool operator== (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool operator== (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _IteratorL, typename _IteratorR >`
`bool operator== (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<class _Dom >`
`_Expr< _BinClos< __equal_to, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __equal_to, typename _Dom::value_type >::result_type > operator== (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< __equal_to, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __equal_to, typename _Dom::value_type >::result_type > operator== (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __equal_to, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __equal_to, typename _Dom1::value_type >::result_type > operator== (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __equal_to, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< __equal_to, typename _Dom::value_type >::result_type > operator== (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`

- `template<class _Dom >`
`_Expr< _BinClos< __equal_to, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< __equal_to, typename _Dom::value_type >::result_type > operator==(const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<typename _T1, typename _T2 >`
`constexpr bool operator==(const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename _OutA1, typename _OutA2, typename... _InA>`
`bool operator==(const scoped_allocator_adaptor< _OutA1, _InA... > &__a, const scoped_allocator_adaptor< _OutA2, _InA... > &__b) noexcept`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`bool operator==(const unique_ptr< _Tp, _Dp > &__x, const unique_ptr< _Up, _Ep > &__y)`
- `template<typename _Tp, typename _Dp >`
`bool operator==(const unique_ptr< _Tp, _Dp > &__x, nullptr_t) noexcept`
- `template<typename _Tp, typename _Dp >`
`bool operator==(nullptr_t, const unique_ptr< _Tp, _Dp > &__x) noexcept`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator==(const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator==(const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Bilter >`
`bool operator==(const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator==(const multimap< _Key, _Tp, _Compare, _Alloc > &__x, const multimap< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
`bool operator==(const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__lhs, const sub_match< __Bi_iter > &__rhs)`
- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
`bool operator==(const sub_match< _Bi_iter > &__lhs, const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__rhs)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool operator==(const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _Iterator >`
`bool operator==(const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _Bi_iter >`
`bool operator==(typename iterator_traits< _Bi_iter >::value_type const * __lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Tp >`
`_Expr< _BinClos< __equal_to, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __equal_to, _Tp >::result_type > operator==(const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __equal_to, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __equal_to, _Tp >::result_type > operator==(const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __equal_to, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __equal_to, _Tp >::result_type > operator==(const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>`
`bool operator==(const __shared_ptr< _Tp1, _Lp > &__a, const __shared_ptr< _Tp2, _Lp > &__b) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator==(const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator==(nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`

- `template<typename _Bi_iter >`
`bool operator== (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const * __rhs)`
- `template<typename _Key, typename _Val, typename _KeyOfValue, typename _Compare, typename _Alloc >`
`bool operator== (const _Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc > &__x, const _Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator== (const map< _Key, _Tp, _Compare, _Alloc > &__x, const map< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Bi_iter >`
`bool operator== (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool operator== (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename _Tp, typename _Alloc >`
`bool operator== (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Bi_iter >`
`bool operator== (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<class _Value, class _Hash, class _Pred, class _Alloc >`
`bool operator== (const unordered_set< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_set< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<class _Value, class _Hash, class _Pred, class _Alloc >`
`bool operator== (const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool operator== (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `template<class _Key, class _Tp, class _Hash, class _Pred, class _Alloc >`
`bool operator== (const unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, const unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > &__y)`
- `template<class _Key, class _Tp, class _Hash, class _Pred, class _Alloc >`
`bool operator== (const unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, const unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`_GLIBCXX_END_NAMESPACE_CXX11 bool operator== (const list< _Tp, _Alloc > &__x, const list< _Tp, _Alloc > &__y)`
- `template<typename _RealType >`
`bool operator== (const std::normal_distribution< _RealType > &__d1, const std::normal_distribution< _RealType > &__d2)`
- `template<typename _Bi_iter, typename _Alloc >`
`bool operator== (const match_results< _Bi_iter, _Alloc > &__m1, const match_results< _Bi_iter, _Alloc > &__m2)`
- `template<typename _Res, typename... _Args>`
`bool operator== (const function< _Res(_Args...)> &__f, nullptr_t) noexcept`
- `template<typename _Res, typename... _Args>`
`bool operator== (nullptr_t, const function< _Res(_Args...)> &__f) noexcept`
- `template<typename _Tp, typename _Alloc >`
`bool operator== (const deque< _Tp, _Alloc > &__x, const deque< _Tp, _Alloc > &__y)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator== (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs) noexcept`
- `template<typename _CharT >`
`__gnu_cxx::enable_if< __is_char< _CharT >::__value, bool >::__type operator== (const basic_string< _CharT > &__lhs, const basic_string< _CharT > &__rhs) noexcept`

- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator== (const _CharT * __lhs, const basic_string< _CharT, _Traits, _Alloc > & __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator== (const basic_string< _CharT, _Traits, _Alloc > & __lhs, const _CharT * __rhs)`
- `bool operator> (thread::id __x, thread::id __y) noexcept`
- `template<typename _Tp, std::size_t _Nm >`
`bool operator> (const array< _Tp, _Nm > & __one, const array< _Tp, _Nm > & __two)`
- `template<typename _Tp, typename _Seq >`
`bool operator> (const stack< _Tp, _Seq > & __x, const stack< _Tp, _Seq > & __y)`
- `template<typename _Tp, typename _Ref, typename _Ptr >`
`bool operator> (const _Deque_iterator< _Tp, _Ref, _Ptr > & __x, const _Deque_iterator< _Tp, _Ref, _Ptr > & __y) noexcept`
- `template<typename _Iterator >`
`bool operator> (const reverse_iterator< _Iterator > & __x, const reverse_iterator< _Iterator > & __y)`
- `template<typename _Tp, typename _RefL, typename _PtrL, typename _RefR, typename _PtrR >`
`bool operator> (const _Deque_iterator< _Tp, _RefL, _PtrL > & __x, const _Deque_iterator< _Tp, _RefR, _PtrR > & __y) noexcept`
- `template<typename _Tp, typename _Seq >`
`bool operator> (const queue< _Tp, _Seq > & __x, const queue< _Tp, _Seq > & __y)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool operator> (const reverse_iterator< _IteratorL > & __x, const reverse_iterator< _IteratorR > & __y)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator> (const shared_ptr< _Tp1 > & __a, const shared_ptr< _Tp2 > & __b) noexcept`
- `template<typename _Tp >`
`bool operator> (const shared_ptr< _Tp > & __a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool operator> (nullptr_t, const shared_ptr< _Tp > & __a) noexcept`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __greater, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __greater, typename _Dom1::value_type >::result_type > operator> (const _Expr< _Dom1, typename _Dom1::value_type > & __v, const _Expr< _Dom2, typename _Dom2::value_type > & __w)`
- `template<class _Dom >`
`_Expr< _BinClos< __greater, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< __greater, typename _Dom::value_type >::result_type > operator> (const _Expr< _Dom, typename _Dom::value_type > & __v, const typename _Dom::value_type & __t)`
- `template<class _Dom >`
`_Expr< _BinClos< __greater, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __greater, typename _Dom::value_type >::result_type > operator> (const valarray< typename _Dom::value_type > & __v, const _Expr< _Dom, typename _Dom::value_type > & __e)`
- `template<class _Dom >`
`_Expr< _BinClos< __greater, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< __greater, typename _Dom::value_type >::result_type > operator> (const _Expr< _Dom, typename _Dom::value_type > & __e, const valarray< typename _Dom::value_type > & __v)`
- `template<class _Dom >`
`_Expr< _BinClos< __greater, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __greater, typename _Dom::value_type >::result_type > operator> (const typename _Dom::value_type & __t, const _Expr< _Dom, typename _Dom::value_type > & __v)`
- `template<typename _T1, typename _T2 >`
`constexpr bool operator> (const pair< _T1, _T2 > & __x, const pair< _T1, _T2 > & __y)`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`bool operator> (const unique_ptr< _Tp, _Dp > & __x, const unique_ptr< _Up, _Ep > & __y)`
- `template<typename _Tp, typename _Dp >`
`bool operator> (const unique_ptr< _Tp, _Dp > & __x, nullptr_t)`

- `template<typename _Tp, typename _Dp >`
`bool operator> (nullptr_t, const unique_ptr< _Tp, _Dp > &__x)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator> (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator> (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Bilter >`
`bool operator> (const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator> (const multimap< _Key, _Tp, _Compare, _Alloc > &__x, const multimap< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
`bool operator> (const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter, class _Ch_traits, class _Ch_alloc >`
`bool operator> (const sub_match< _Bi_iter > &__lhs, const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__rhs)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool operator> (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _Iterator >`
`bool operator> (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _Tp >`
`__Expr< __BinClos< __greater, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __greater, _Tp >::result_type > operator> (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`__Expr< __BinClos< __greater, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __greater, _Tp >::result_type > operator> (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`__Expr< __BinClos< __greater, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __greater, _Tp >::result_type > operator> (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Bi_iter >`
`bool operator> (typename iterator_traits< _Bi_iter >::value_type const *__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>`
`bool operator> (const __shared_ptr< _Tp1, _Lp > &__a, const __shared_ptr< _Tp2, _Lp > &__b) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator> (const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator> (nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`
- `template<typename _Bi_iter >`
`bool operator> (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const *__rhs)`
- `template<typename _Key, typename _Val, typename _KeyOfValue, typename _Compare, typename _Alloc >`
`bool operator> (const _Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc > &__x, const _Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator> (const map< _Key, _Tp, _Compare, _Alloc > &__x, const map< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Bi_iter >`
`bool operator> (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`

- `template<typename... _TElements, typename... _UElements>`
`constexpr bool operator> (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename _Tp, typename _Alloc >`
`bool operator> (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Bi_iter >`
`bool operator> (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator> (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool operator> (const list< _Tp, _Alloc > &__x, const list< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool operator> (const deque< _Tp, _Alloc > &__x, const deque< _Tp, _Alloc > &__y)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator> (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator> (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator> (const _CharT *__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `bool operator>= (thread::id __x, thread::id __y) noexcept`
- `template<typename _Tp, std::size_t _Nm>`
`bool operator>= (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _Tp, typename _Seq >`
`bool operator>= (const stack< _Tp, _Seq > &__x, const stack< _Tp, _Seq > &__y)`
- `template<typename _Iterator >`
`bool operator>= (const reverse_iterator< _Iterator > &__x, const reverse_iterator< _Iterator > &__y)`
- `template<typename _Tp, typename _Ref, typename _Ptr >`
`bool operator>= (const _Deque_iterator< _Tp, _Ref, _Ptr > &__x, const _Deque_iterator< _Tp, _Ref, _Ptr > &__y) noexcept`
- `template<typename _Tp, typename _RefL, typename _PtrL, typename _RefR, typename _PtrR >`
`bool operator>= (const _Deque_iterator< _Tp, _RefL, _PtrL > &__x, const _Deque_iterator< _Tp, _RefR, _PtrR > &__y) noexcept`
- `template<typename _Tp, typename _Seq >`
`bool operator>= (const queue< _Tp, _Seq > &__x, const queue< _Tp, _Seq > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool operator>= (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __greater_equal, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __greater_equal, typename _Dom1::value_type >::result_type > operator>= (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __greater_equal, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __greater_equal, typename _Dom::value_type >::result_type > operator>= (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< __greater_equal, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< __greater_equal, typename _Dom::value_type >::result_type > operator>= (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __greater_equal, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __greater_equal, typename _Dom::value_type >::result_type > operator>= (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`

- `template<class _Dom >`
`_Expr< __BinClos< __greater_equal, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< __greater_equal, typename _Dom::value_type >::result_type > operator>= (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<typename _Tp1, typename _Tp2 >`
`bool operator>= (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool operator>= (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool operator>= (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _T1, typename _T2 >`
`constexpr bool operator>= (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`bool operator>= (const unique_ptr< _Tp, _Dp > &__x, const unique_ptr< _Up, _Ep > &__y)`
- `template<typename _Tp, typename _Dp >`
`bool operator>= (const unique_ptr< _Tp, _Dp > &__x, nullptr_t)`
- `template<typename _Tp, typename _Dp >`
`bool operator>= (nullptr_t, const unique_ptr< _Tp, _Dp > &__x)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator>= (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`bool operator>= (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Bilter >`
`bool operator>= (const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator>= (const multimap< _Key, _Tp, _Compare, _Alloc > &__x, const multimap< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
`bool operator>= (const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter, class _Ch_traits, class _Ch_alloc >`
`bool operator>= (const sub_match< _Bi_iter > &__lhs, const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__rhs)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool operator>= (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _Iterator >`
`bool operator>= (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _Tp >`
`_Expr< __BinClos< __greater_equal, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __greater_equal, _Tp >::result_type > operator>= (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< __BinClos< __greater_equal, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __greater_equal, _Tp >::result_type > operator>= (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< __BinClos< __greater_equal, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __greater_equal, _Tp >::result_type > operator>= (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Bi_iter >`
`bool operator>= (typename iterator_traits< _Bi_iter >::value_type const *__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>`
`bool operator>= (const __shared_ptr< _Tp1, _Lp > &__a, const __shared_ptr< _Tp2, _Lp > &__b) noexcept`

- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator>= (const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool operator>= (nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`
- `template<typename _Bi_iter >`
`bool operator>= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const * __rhs)`
- `template<typename _Key, typename _Val, typename _KeyOfValue, typename _Compare, typename _Alloc >`
`bool operator>= (const _Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc > &__x, const _Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`bool operator>= (const map< _Key, _Tp, _Compare, _Alloc > &__x, const map< _Key, _Tp, _Compare, _Alloc > &__y)`
- `template<typename _Bi_iter >`
`bool operator>= (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool operator>= (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename _Tp, typename _Alloc >`
`bool operator>= (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Bi_iter >`
`bool operator>= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator>= (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool operator>= (const list< _Tp, _Alloc > &__x, const list< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool operator>= (const deque< _Tp, _Alloc > &__x, const deque< _Tp, _Alloc > &__y)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator>= (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator>= (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const _CharT * __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool operator>= (const _CharT * __lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__is, _Resetiosflags __f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__is, _Setiosflags __f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__is, _Setbase __f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__is, _Setfill< _CharT > __f)`
- `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, linear_↵ congruential_engine< _UIntType, __a, __c, __m > &__lcr)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__is, _Setprecision __f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__is, _Setw __f)`

- `template<typename _CharT, typename _Traits, typename _MoneyT >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > & __is, _Get_money< _MoneyT > __f)`
- `template<class _Dom >`
`_Expr< _BinClos< __shift_right, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __shift_right, typename _Dom::value_type >::result_type > operator>> (const valarray< typename _Dom::value_type > & __v, const _Expr< _Dom, typename _Dom::value_type > & __e)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __shift_right, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __shift_right, typename _Dom1::value_type >::result_type > operator>> (const _Expr< _Dom1, typename _Dom1::value_type > & __v, const _Expr< _Dom2, typename _Dom2::value_type > & __w)`
- `template<class _Dom >`
`_Expr< _BinClos< __shift_right, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< __shift_right, typename _Dom::value_type >::result_type > operator>> (const _Expr< _Dom, typename _Dom::value_type > & __v, const typename _Dom::value_type & __t)`
- `template<class _Dom >`
`_Expr< _BinClos< __shift_right, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< __shift_right, typename _Dom::value_type >::result_type > operator>> (const _Expr< _Dom, typename _Dom::value_type > & __e, const valarray< typename _Dom::value_type > & __v)`
- `template<class _Dom >`
`_Expr< _BinClos< __shift_right, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __shift_right, typename _Dom::value_type >::result_type > operator>> (const typename _Dom::value_type & __t, const _Expr< _Dom, typename _Dom::value_type > & __v)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > & __is, _Get_time< _CharT > __f)`
- `template<typename _Tp, typename _CharT, class _Traits >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > & __is, complex< _Tp > & __x)`
- `template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f > & __x)`
- `template<typename _UIntType, size_t __w, size_t __s, size_t __r, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, subtract_with_carry_engine< _UIntType, __w, __s, __r > & __x)`
- `template<typename _RandomNumberEngine, size_t __p, size_t __r, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, discard_block_engine< _RandomNumberEngine, __p, __r > & __x)`
- `template<typename _RandomNumberEngine, size_t __k, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, shuffle_order_engine< _RandomNumberEngine, __k > & __x)`
- `template<typename _CharT, typename _Traits, typename _Tp >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > & __is, _Tp & __x)`
- `template<typename _Tp >`
`_Expr< _BinClos< __shift_right, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __shift_right, _Tp >::result_type > operator>> (const _Tp & __t, const valarray< _Tp > & __v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __shift_right, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __shift_right, _Tp >::result_type > operator>> (const valarray< _Tp > & __v, const valarray< _Tp > & __w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __shift_right, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __shift_right, _Tp >::result_type > operator>> (const valarray< _Tp > & __v, const _Tp & __t)`

- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, negative←`
`_binomial_distribution< _IntType > & __x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, poisson←`
`_distribution< _IntType > & __x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, std::uniform←`
`int_distribution< _IntType > & __x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, binomial←`
`_distribution< _IntType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, std::uniform←`
`real_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, normal←`
`distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, lognormal←`
`distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, chi←`
`squared_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, fisher_f←`
`distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, student←`
`_t_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, gamma←`
`_distribution< _RealType > & __x)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > & __is, __gnu_cxx::__←`
`versa_string< _CharT, _Traits, _Alloc, _Base > & __str)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, discrete←`
`_distribution< _IntType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, std::←`
`cauchy_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, piecewise←`
`_constant_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, piecewise←`
`_linear_distribution< _RealType > & __x)`
- `template<typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, std::←`
`bernoulli_distribution & __x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, std::←`
`geometric_distribution< _IntType > & __x)`

- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, std::←`
`::exponential_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, std::←`
`::weibull_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, std::←`
`::extreme_value_distribution< _RealType > &__x)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__is, basic_string< _←`
`CharT, _Traits, _Alloc > &__str)`
- `template<>`
`basic_istream< char > & operator>> (basic_istream< char > &__is, basic_string< char > &__str)`
- `constexpr _ios_Fmtflags operator^ (_ios_Fmtflags __a, _ios_Fmtflags __b)`
- `constexpr _ios_Openmode operator^ (_ios_Openmode __a, _ios_Openmode __b)`
- `constexpr launch operator^ (launch __x, launch __y)`
- `constexpr _ios_ostate operator^ (_ios_ostate __a, _ios_ostate __b)`
- `template<class _Dom >`
`_Expr< _BinClos< __bitwise_xor, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun<`
`__bitwise_xor, typename _Dom::value_type >::result_type > operator^ (const _Expr< _Dom, typename _←`
`Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __bitwise_xor, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __bitwise_xor, typename`
`_Dom1::value_type >::result_type > operator^ (const _Expr< _Dom1, typename _Dom1::value_type > &__v,`
`const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __bitwise_xor, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__bitwise_xor, typename _Dom::value_type >::result_type > operator^ (const typename _Dom::value_type &←`
`__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __bitwise_xor, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __←`
`fun< __bitwise_xor, typename _Dom::value_type >::result_type > operator^ (const _Expr< _Dom, typename`
`_Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __bitwise_xor, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__bitwise_xor, typename _Dom::value_type >::result_type > operator^ (const valarray< typename _Dom←`
`::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _BinClos< __bitwise_xor, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __bitwise_xor, _Tp >←`
`::result_type > operator^ (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __bitwise_xor, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __bitwise_xor, _Tp >←`
`::result_type > operator^ (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __bitwise_xor, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __bitwise_xor, _Tp >←`
`::result_type > operator^ (const _Tp &__t, const valarray< _Tp > &__v)`
- `const _ios_Fmtflags & operator^= (_ios_Fmtflags &__a, _ios_Fmtflags __b)`
- `const _ios_Openmode & operator^= (_ios_Openmode &__a, _ios_Openmode __b)`
- `launch & operator^= (launch &__x, launch __y)`
- `const _ios_ostate & operator^= (_ios_ostate &__a, _ios_ostate __b)`
- `constexpr memory_order operator| (memory_order __m, __memory_order_modifier __mod)`
- `constexpr _ios_Fmtflags operator| (_ios_Fmtflags __a, _ios_Fmtflags __b)`

- constexpr `_los_Openmode operator|` (`_los_Openmode __a, _los_Openmode __b`)
- constexpr `launch operator|` (`launch __x, launch __y`)
- constexpr `_los_losestate operator|` (`_los_losestate __a, _los_losestate __b`)
- template<class `_Dom1`, class `_Dom2` >
`_Expr< _BinClos< __bitwise_or, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __bitwise_or, typename
_Dom1::value_type >::result_type > operator|` (`const _Expr< _Dom1, typename _Dom1::value_type > &__v,`
`const _Expr< _Dom2, typename _Dom2::value_type > &__w`)
- template<class `_Dom` >
`_Expr< _BinClos< __bitwise_or, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun<
__bitwise_or, typename _Dom::value_type >::result_type > operator|` (`const typename _Dom::value_type &__t,`
`const _Expr< _Dom, typename _Dom::value_type > &__v`)
- template<class `_Dom` >
`_Expr< _BinClos< __bitwise_or, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun<
__bitwise_or, typename _Dom::value_type >::result_type > operator|` (`const _Expr< _Dom, typename _Dom<←
::value_type > &__e, const valarray< typename _Dom::value_type > &__v`)
- template<class `_Dom` >
`_Expr< _BinClos< __bitwise_or, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __<←
fun< __bitwise_or, typename _Dom::value_type >::result_type > operator|` (`const valarray< typename _Dom<←
::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e`)
- template<class `_Dom` >
`_Expr< _BinClos< __bitwise_or, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun<
__bitwise_or, typename _Dom::value_type >::result_type > operator|` (`const _Expr< _Dom, typename _Dom<←
::value_type > &__v, const typename _Dom::value_type &__t`)
- template<typename `_Tp` >
`_Expr< _BinClos< __bitwise_or, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __bitwise_or, _Tp >←
::result_type > operator|` (`const valarray< _Tp > &__v, const valarray< _Tp > &__w`)
- template<typename `_Tp` >
`_Expr< _BinClos< __bitwise_or, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __bitwise_or, _Tp >←
::result_type > operator|` (`const _Tp &__t, const valarray< _Tp > &__v`)
- template<typename `_Tp` >
`_Expr< _BinClos< __bitwise_or, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __bitwise_or, _Tp >←
::result_type > operator|` (`const valarray< _Tp > &__v, const _Tp &__t`)
- const `_los_Fmtflags & operator|=` (`_los_Fmtflags &__a, _los_Fmtflags __b`)
- const `_los_Openmode & operator|=` (`_los_Openmode &__a, _los_Openmode __b`)
- `launch & operator|=` (`launch &__x, launch __y`)
- const `_los_losestate & operator|=` (`_los_losestate &__a, _los_losestate __b`)
- template<class `_Dom1`, class `_Dom2` >
`_Expr< _BinClos< __logical_or, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __logical_or, typename
_Dom1::value_type >::result_type > operator||` (`const _Expr< _Dom1, typename _Dom1::value_type > &__v,`
`const _Expr< _Dom2, typename _Dom2::value_type > &__w`)
- template<class `_Dom` >
`_Expr< _BinClos< __logical_or, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun<
__logical_or, typename _Dom::value_type >::result_type > operator||` (`const typename _Dom::value_type &__t, const
_Expr< _Dom, typename _Dom::value_type > &__v`)
- template<class `_Dom` >
`_Expr< _BinClos< __logical_or, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun<
__logical_or, typename _Dom::value_type >::result_type > operator||` (`const _Expr< _Dom, typename _Dom<←
::value_type > &__v, const typename _Dom::value_type &__t`)
- template<class `_Dom` >
`_Expr< _BinClos< __logical_or, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __<←
logical_or, typename _Dom::value_type >::result_type > operator||` (`const valarray< typename _Dom::value<←
type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e`)

- `template<class _Dom >`
`_Expr< _BinClos< __logical_or, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun<`
`__logical_or, typename _Dom::value_type >::result_type > operator|| (const _Expr< _Dom, typename _Dom::`
`value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __logical_or, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __logical_or, _Tp >::`
`result_type > operator|| (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __logical_or, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __logical_or, _Tp >::`
`result_type > operator|| (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __logical_or, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __logical_or, _Tp >::`
`result_type > operator|| (const valarray< _Tp > &__v, const _Tp &__t)`
- `constexpr _ios_Fmtflags operator~ (_ios_Fmtflags __a)`
- `constexpr _ios_Openmode operator~ (_ios_Openmode __a)`
- `constexpr launch operator~ (launch __x)`
- `constexpr _ios_istate operator~ (_ios_istate __a)`
- `template<typename _RAIter >`
`void partial_sort (_RAIter, _RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`void partial_sort (_RAIter, _RAIter, _RAIter, _Compare)`
- `template<typename _RandomAccessIterator >`
`void partial_sort (_RandomAccessIterator __first, _RandomAccessIterator __middle, _RandomAccessIterator __`
`last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void partial_sort (_RandomAccessIterator __first, _RandomAccessIterator __middle, _RandomAccessIterator __`
`last, _Compare __comp)`
- `template<typename _Iter, typename _RAIter >`
`_RAIter partial_sort_copy (_Iter, _Iter, _RAIter, _RAIter)`
- `template<typename _Iter, typename _RAIter, typename _Compare >`
`_RAIter partial_sort_copy (_Iter, _Iter, _RAIter, _RAIter, _Compare)`
- `template<typename _InputIterator, typename _RandomAccessIterator >`
`_RandomAccessIterator partial_sort_copy (_InputIterator __first, _InputIterator __last, _RandomAccessIterator`
`__result_first, _RandomAccessIterator __result_last)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _Compare >`
`_RandomAccessIterator partial_sort_copy (_InputIterator __first, _InputIterator __last, _RandomAccessIterator`
`__result_first, _RandomAccessIterator __result_last, _Compare __comp)`
- `template<typename _InputIterator, typename _OutputIterator >`
`_OutputIterator partial_sum (_InputIterator __first, _InputIterator __last, _OutputIterator __result)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator partial_sum (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Binary`
`Operation __binary_op)`
- `template<typename _BIter, typename _Predicate >`
`_BIter partition (_BIter, _BIter, _Predicate)`
- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _Iter, typename _OIter1, typename _OIter2, typename _Predicate >`
`pair< _OIter1, _OIter2 > partition_copy (_Iter, _Iter, _OIter1, _OIter2, _Predicate)`
- `template<typename _InputIterator, typename _OutputIterator1, typename _OutputIterator2, typename _Predicate >`
`pair< _OutputIterator1, _OutputIterator2 > partition_copy (_InputIterator __first, _InputIterator __last, _Output`
`Iterator1 __out_true, _OutputIterator2 __out_false, _Predicate __pred)`
- `template<typename _Filter, typename _Predicate >`
`_Filter partition_point (_Filter, _Filter, _Predicate)`

- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator partition_point (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _Tp >`
`complex< _Tp > polar (const _Tp &, const _Tp &=0)`
- `template<typename _RandomAccessIterator >`
`void pop_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void pop_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RAIter >`
`void pop_heap (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`void pop_heap (_RAIter, _RAIter, _Compare)`
- `template<typename _Tp >`
`complex< _Tp > pow (const complex< _Tp > &, int)`
- `template<typename _Tp >`
`complex< _Tp > pow (const complex< _Tp > &, const _Tp &)`
- `template<typename _Tp >`
`complex< _Tp > pow (const complex< _Tp > &, const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > pow (const _Tp &, const complex< _Tp > &)`
- `constexpr float pow (float __x, float __y)`
- `constexpr long double pow (long double __x, long double __y)`
- `template<typename _Tp, typename _Up >`
`constexpr gnu_cxx::promote_2< _Tp, _Up >::__type pow (_Tp __x, _Up __y)`
- `template<typename _Tp >`
`_Expr< _BinClos< _Pow, _ValArray, _ValArray, _Tp, _Tp >, _Tp > pow (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< _Pow, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename _Dom::value_type > pow (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< _Pow, _ValArray, _Constant, _Tp, _Tp >, _Tp > pow (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< _Pow, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_type > pow (const valarray< typename _Dom::valarray > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _BinClos< _Pow, _Constant, _ValArray, _Tp, _Tp >, _Tp > pow (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< _Pow, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_type > pow (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< _Pow, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename _Dom::value_type > pow (const _Expr< _Dom, typename _Dom::value_type > &__e, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< _Pow, _Expr, _Expr, _Dom1, _Dom2 >, typename _Dom1::value_type > pow (const _Expr< _Dom1, typename _Dom1::value_type > &__e1, const _Expr< _Dom2, typename _Dom2::value_type > &__e2)`

- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > pow (const std::complex< _Tp >`
`&__x, const _Up &__y)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > pow (const _Tp &__x, const std::complex< _Up > &__y)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > pow (const std::complex< _Tp >`
`&__x, const std::complex< _Up > &__y)`
- `template<typename _BidirectionalIterator >`
`_BidirectionalIterator prev (_BidirectionalIterator __x, typename iterator_traits< _BidirectionalIterator >::`
`difference_type __n=1)`
- `template<typename _Blter >`
`bool prev_permutation (_Blter, _Blter)`
- `template<typename _Blter, typename _Compare >`
`bool prev_permutation (_Blter, _Blter, _Compare)`
- `template<typename _BidirectionalIterator >`
`bool prev_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`bool prev_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare __comp)`
- `template<typename _Tp >`
`std::complex< _Tp > proj (const std::complex< _Tp > &)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type proj (_Tp __x)`
- `template<typename _Arg, typename _Result >`
`pointer_to_unary_function< _Arg, _Result > ptr_fun (_Result(*__x)(_Arg))`
- `template<typename _Arg1, typename _Arg2, typename _Result >`
`pointer_to_binary_function< _Arg1, _Arg2, _Result > ptr_fun (_Result(*__x)(_Arg1, _Arg2))`
- `template<typename _RandomAccessIterator >`
`void push_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void push_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RAIter >`
`void push_heap (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`void push_heap (_RAIter, _RAIter, _Compare)`
- `template<typename _MoneyT >`
`_Put_money< _MoneyT > put_money (const _MoneyT &__mon, bool __intl=false)`
- `template<typename _CharT >`
`_Put_time< _CharT > put_time (const std::tm *__tmb, const _CharT *__fmt)`
- `template<typename _CharT >`
`auto quoted (const _CharT *__string, _CharT __delim=_CharT(""), _CharT __escape = _CharT("\\"))`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`auto quoted (const basic_string< _CharT, _Traits, _Alloc > &__string, _CharT __delim=_CharT(""), _CharT __escape = _CharT("\\"))`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`auto quoted (basic_string< _CharT, _Traits, _Alloc > &__string, _CharT __delim=_CharT(""), _CharT __escape = _CharT("\\"))`
- `template<typename _RAIter >`
`void random_shuffle (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Generator >`
`void random_shuffle (_RAIter, _RAIter, _Generator &&)`

- `template<typename _RandomAccessIterator, typename _RandomNumberGenerator >`
`void random_shuffle (_RandomAccessIterator __first, _RandomAccessIterator __last, _RandomNumberGenerator &&__rand)`
- `template<typename _Container >`
`auto rbegin (_Container &__cont) -> decltype(__cont.rbegin())`
- `template<typename _Container >`
`auto rbegin (const _Container &__cont) -> decltype(__cont.rbegin())`
- `template<typename _Tp, size_t _Nm>`
`reverse_iterator< _Tp * > rbegin (_Tp(&__arr)[_Nm])`
- `template<typename _Tp >`
`reverse_iterator< const _Tp * > rbegin (initializer_list< _Tp > __il)`
- `template<typename _Tp >`
`constexpr _Tp real (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type real (_Tp __x)`
- `template<typename _Filter, typename _Tp >`
`_Filter remove (_Filter, _Filter, const _Tp &)`
- `template<typename _ForwardIterator, typename _Tp >`
`_ForwardIterator remove (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value)`
- `template<typename _Iiter, typename _Oiter, typename _Tp >`
`_Oiter remove_copy (_Iiter, _Iiter, _Oiter, const _Tp &)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Tp >`
`_OutputIterator remove_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result, const _Tp &__value)`
- `template<typename _Iiter, typename _Oiter, typename _Predicate >`
`_Oiter remove_copy_if (_Iiter, _Iiter, _Oiter, _Predicate)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate >`
`_OutputIterator remove_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Predicate __pred)`
- `template<typename _Filter, typename _Predicate >`
`_Filter remove_if (_Filter, _Filter, _Predicate)`
- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator remove_if (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _Container >`
`auto rend (_Container &__cont) -> decltype(__cont.rend())`
- `template<typename _Container >`
`auto rend (const _Container &__cont) -> decltype(__cont.rend())`
- `template<typename _Tp, size_t _Nm>`
`reverse_iterator< _Tp * > rend (_Tp(&__arr)[_Nm])`
- `template<typename _Tp >`
`reverse_iterator< const _Tp * > rend (initializer_list< _Tp > __il)`
- `template<typename _Filter, typename _Tp >`
`void replace (_Filter, _Filter, const _Tp &, const _Tp &)`
- `template<typename _ForwardIterator, typename _Tp >`
`void replace (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__old_value, const _Tp &__new_value)`
- `template<typename _Iiter, typename _Oiter, typename _Tp >`
`_Oiter replace_copy (_Iiter, _Iiter, _Oiter, const _Tp &, const _Tp &)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Tp >`
`_OutputIterator replace_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result, const _Tp &__old_value, const _Tp &__new_value)`
- `template<typename _Iiter, typename _Oiter, typename _Predicate, typename _Tp >`
`_Oiter replace_copy_if (_Iiter, _Iiter, _Oiter, _Predicate, const _Tp &)`

- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate, typename _Tp >`
`_OutputIterator replace_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Predicate`
`__pred, const _Tp & __new_value)`
- `template<typename _Filter, typename _Predicate, typename _Tp >`
`void replace_if (_Filter, _Filter, _Predicate, const _Tp &)`
- `template<typename _ForwardIterator, typename _Predicate, typename _Tp >`
`void replace_if (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred, const _Tp & __new_value)`
- `_Resettiosflags resettiosflags (ios_base::fmtflags __mask)`
- `void rethrow_exception (exception_ptr) __attribute__((__noreturn__))`
- `template<typename _Ex >`
`void rethrow_if_nested (const _Ex & __ex)`
- `template<typename _Tp >`
`void return_temporary_buffer (_Tp * __p)`
- `template<typename _Blter >`
`void reverse (_Blter, _Blter)`
- `template<typename _BidirectionalIterator >`
`void reverse (_BidirectionalIterator __first, _BidirectionalIterator __last)`
- `template<typename _Blter, typename _Olter >`
`_Olter reverse_copy (_Blter, _Blter, _Olter)`
- `template<typename _BidirectionalIterator, typename _OutputIterator >`
`_OutputIterator reverse_copy (_BidirectionalIterator __first, _BidirectionalIterator __last, _OutputIterator __result)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type riemann_zeta (_Tp __s)`
- `float riemann_zetaf (float __s)`
- `long double riemann_zetal (long double __s)`
- `ios_base & right (ios_base & __base)`
- `template<typename _Filter, typename _Olter >`
`_Olter rotate_copy (_Filter, _Filter, _Filter, _Olter)`
- `template<typename _ForwardIterator, typename _OutputIterator >`
`_OutputIterator rotate_copy (_ForwardIterator __first, _ForwardIterator __middle, _ForwardIterator __last, ↵`
`_OutputIterator __result)`
- `ios_base & scientific (ios_base & __base)`
- `template<typename _Filter1, typename _Filter2 >`
`_Filter1 search (_Filter1, _Filter1, _Filter2, _Filter2)`
- `template<typename _Filter1, typename _Filter2, typename _BinaryPredicate >`
`_Filter1 search (_Filter1, _Filter1, _Filter2, _Filter2, _BinaryPredicate)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`_ForwardIterator1 search (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, ↵`
`_ForwardIterator2 __last2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`_ForwardIterator1 search (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, ↵`
`_ForwardIterator2 __last2, _BinaryPredicate __predicate)`
- `template<typename _Filter, typename _Size, typename _Tp >`
`_Filter search_n (_Filter, _Filter, _Size, const _Tp &)`
- `template<typename _Filter, typename _Size, typename _Tp, typename _BinaryPredicate >`
`_Filter search_n (_Filter, _Filter, _Size, const _Tp &, _BinaryPredicate)`
- `template<typename _ForwardIterator, typename _Integer, typename _Tp >`
`_ForwardIterator search_n (_ForwardIterator __first, _ForwardIterator __last, _Integer __count, const _Tp & __val)`
- `template<typename _ForwardIterator, typename _Integer, typename _Tp, typename _BinaryPredicate >`
`_ForwardIterator search_n (_ForwardIterator __first, _ForwardIterator __last, _Integer __count, const _Tp & __val,`
`_BinaryPredicate __binary_pred)`
- `template<typename _Ilter1, typename _Ilter2, typename _Olter >`
`_Olter set_difference (_Ilter1, _Ilter1, _Ilter2, _Ilter2, _Olter)`

- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`_OIter set_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`_OutputIterator set_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator set_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter set_intersection (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`_OIter set_intersection (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`_OutputIterator set_intersection (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator set_intersection (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `new_handler set_new_handler (new_handler) throw ()`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter set_symmetric_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`_OIter set_symmetric_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`_OutputIterator set_symmetric_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator set_symmetric_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `terminate_handler set_terminate (terminate_handler) noexcept`
- `unexpected_handler set_unexpected (unexpected_handler) noexcept`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`_OIter set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`_OutputIterator set_union (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator set_union (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `_Setbase setbase (int __base)`
- `template<typename _CharT >`
`_Setfill< _CharT > setfill (_CharT __c)`
- `_Setiosflags setiosflags (ios_base::fmtflags __mask)`
- `_Setprecision setprecision (int __n)`
- `_Setw setw (int __n)`
- `ios_base & showbase (ios_base & __base)`
- `ios_base & showpoint (ios_base & __base)`
- `ios_base & showpos (ios_base & __base)`
- `template<typename _RAIter, typename _UGenerator >`
`void shuffle (_RAIter, _RAIter, _UGenerator &&)`

- `template<typename _RandomAccessIterator, typename _UniformRandomNumberGenerator >`
`void shuffle (_RandomAccessIterator __first, _RandomAccessIterator __last, _UniformRandomNumberGenerator`
`&& __g)`
- `template<typename _Tp >`
`complex< _Tp > sin (const complex< _Tp > &)`
- `template<class _Dom >`
`_Expr< _UnClos< _Sin, _Expr, _Dom >, typename _Dom::value_type > sin (const _Expr< _Dom, typename`
`_Dom::value_type > & __e)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Sin, _ValArray, _Tp >, _Tp > sin (const valarray< _Tp > & __v)`
- `constexpr float sin (float __x)`
- `constexpr long double sin (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type sin (_Tp __x)`
- `template<typename _Tp >`
`complex< _Tp > sinh (const complex< _Tp > &)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Sinh, _ValArray, _Tp >, _Tp > sinh (const valarray< _Tp > & __v)`
- `template<class _Dom >`
`_Expr< _UnClos< _Sinh, _Expr, _Dom >, typename _Dom::value_type > sinh (const _Expr< _Dom, typename`
`_Dom::value_type > & __e)`
- `constexpr float sinh (float __x)`
- `constexpr long double sinh (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type sinh (_Tp __x)`
- `ios_base & skipws (ios_base & __base)`
- `template<typename _RAIter >`
`void sort (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`void sort (_RAIter, _RAIter, _Compare)`
- `template<typename _RandomAccessIterator >`
`void sort (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`void sort_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void sort_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RAIter >`
`void sort_heap (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`void sort_heap (_RAIter, _RAIter, _Compare)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type sph_bessel (unsigned int __n, _Tp __x)`
- `float sph_besself (unsigned int __n, float __x)`
- `long double sph_bessell (unsigned int __n, long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type sph_legendre (unsigned int __l, unsigned int __m, _Tp __theta)`
- `float sph_legendref (unsigned int __l, unsigned int __m, float __theta)`
- `long double sph_legendrel (unsigned int __l, unsigned int __m, long double __theta)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type sph_neumann (unsigned int __n, _Tp __x)`

- float [sph_neumannf](#) (unsigned int __n, float __x)
- long double [sph_neumannl](#) (unsigned int __n, long double __x)
- template<typename _Tp >
[complex](#)< _Tp > [sqrt](#) (const [complex](#)< _Tp > &)
- template<typename _Tp >
_Expr< _UnClos< _Sqrt, _ValArray, _Tp >, _Tp > [sqrt](#) (const [valarray](#)< _Tp > & __v)
- template<class _Dom >
_Expr< _UnClos< _Sqrt, _Expr, _Dom >, typename _Dom::value_type > [sqrt](#) (const _Expr< _Dom, typename _Dom::value_type > & __e)
- constexpr float [sqrt](#) (float __x)
- constexpr long double [sqrt](#) (long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::enable_if< __is_integer< _Tp >::__value, double >::__type [sqrt](#) (_Tp __x)
- template<typename _Biter, typename _Predicate >
_Biter [stable_partition](#) (_Biter, _Biter, _Predicate)
- template<typename _ForwardIterator, typename _Predicate >
_ForwardIterator [stable_partition](#) (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)
- template<typename _RAIter >
void [stable_sort](#) (_RAIter, _RAIter)
- template<typename _RAIter, typename _Compare >
void [stable_sort](#) (_RAIter, _RAIter, _Compare)
- template<typename _RandomAccessIterator >
void [stable_sort](#) (_RandomAccessIterator __first, _RandomAccessIterator __last)
- template<typename _RandomAccessIterator, typename _Compare >
void [stable_sort](#) (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)
- template<typename _Tp, typename _Tp1 >
[shared_ptr](#)< _Tp > [static_pointer_cast](#) (const [shared_ptr](#)< _Tp1 > & __r) noexcept
- template<typename _Tp, typename _Tp1, _Lock_policy _Lp >
__shared_ptr< _Tp, _Lp > [static_pointer_cast](#) (const __shared_ptr< _Tp1, _Lp > & __r) noexcept
- char * [strchr](#) (char * __s, int __n)
- char * [strpbrk](#) (char * __s1, const char * __s2)
- char * [strrchr](#) (char * __s, int __n)
- char * [strstr](#) (char * __s1, const char * __s2)
- void [swap](#) (_Bit_reference __x, _Bit_reference __y) noexcept
- void [swap](#) (_Bit_reference __x, bool & __y) noexcept
- void [swap](#) (bool & __x, _Bit_reference __y) noexcept
- template<typename _Tp >
enable_if< __and< is_move_constructible< _Tp >, is_move_assignable< _Tp >::__value >::__type [swap](#) (_Tp & __a, _Tp & __b) noexcept(__and< is_nothrow_move_constructible< _Tp >, is_nothrow_move_assignable< _Tp >::__value)
- template<typename _Tp, size_t _Nm >
enable_if< __is_swappable< _Tp >::__value >::__type [swap](#) (_Tp(& __a)[_Nm], _Tp(& __b)[_Nm]) noexcept(__is_nothrow_swappable< _Tp >::__value)
- void [swap](#) (thread & __x, thread & __y) noexcept
- template<typename _Tp, std::size_t _Nm >
void [swap](#) (array< _Tp, _Nm > & __one, array< _Tp, _Nm > & __two) noexcept(noexcept(__one.swap(__two)))
- template<typename _Tp, typename _Seq >
void [swap](#) (stack< _Tp, _Seq > & __x, stack< _Tp, _Seq > & __y) noexcept(noexcept(__x.swap(__y)))
- template<typename _Tp, typename _Seq >
void [swap](#) (queue< _Tp, _Seq > & __x, queue< _Tp, _Seq > & __y) noexcept(noexcept(__x.swap(__y)))
- template<typename _Mutex >
void [swap](#) (unique_lock< _Mutex > & __x, unique_lock< _Mutex > & __y) noexcept

- `template<typename _Tp >`
`void swap (shared_ptr< _Tp > &__a, shared_ptr< _Tp > &__b) noexcept`
- `template<typename _T1, typename _T2 >`
`void swap (pair< _T1, _T2 > &__x, pair< _T1, _T2 > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Tp >`
`void swap (weak_ptr< _Tp > &__a, weak_ptr< _Tp > &__b) noexcept`
- `template<typename _Tp, typename _Sequence, typename _Compare >`
`void swap (priority_queue< _Tp, _Sequence, _Compare > &__x, priority_queue< _Tp, _Sequence, _Compare > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Tp, typename _Dp >`
`void swap (unique_ptr< _Tp, _Dp > &__x, unique_ptr< _Tp, _Dp > &__y) noexcept`
- `template<class _CharT, class _Traits, class _Allocator >`
`void swap (basic_stringbuf< _CharT, _Traits, _Allocator > &__x, basic_stringbuf< _CharT, _Traits, _Allocator > &__y)`
- `template<class _CharT, class _Traits, class _Allocator >`
`void swap (basic_istream< _CharT, _Traits, _Allocator > &__x, basic_istream< _CharT, _Traits, _Allocator > &__y)`
- `template<class _CharT, class _Traits, class _Allocator >`
`void swap (basic_ostringstream< _CharT, _Traits, _Allocator > &__x, basic_ostringstream< _CharT, _Traits, _Allocator > &__y)`
- `template<class _CharT, class _Traits, class _Allocator >`
`void swap (basic_stringstream< _CharT, _Traits, _Allocator > &__x, basic_stringstream< _CharT, _Traits, _Allocator > &__y)`
- `template<typename _Ch_type, typename _Rx_traits >`
`void swap (basic_regex< _Ch_type, _Rx_traits > &__lhs, basic_regex< _Ch_type, _Rx_traits > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`void swap (multiset< _Key, _Compare, _Alloc > &__x, multiset< _Key, _Compare, _Alloc > &__y) noexcept(/*conditional */)`
- `template<typename _Key, typename _Compare, typename _Alloc >`
`void swap (set< _Key, _Compare, _Alloc > &__x, set< _Key, _Compare, _Alloc > &__y) noexcept(/*conditional */)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`void swap (multimap< _Key, _Tp, _Compare, _Alloc > &__x, multimap< _Key, _Tp, _Compare, _Alloc > &__y) noexcept(/*conditional */)`
- `template<class _CharT, class _Traits >`
`void swap (basic_filebuf< _CharT, _Traits > &__x, basic_filebuf< _CharT, _Traits > &__y)`
- `template<class _CharT, class _Traits >`
`void swap (basic_ifstream< _CharT, _Traits > &__x, basic_ifstream< _CharT, _Traits > &__y)`
- `template<class _CharT, class _Traits >`
`void swap (basic_ofstream< _CharT, _Traits > &__x, basic_ofstream< _CharT, _Traits > &__y)`
- `template<class _CharT, class _Traits >`
`void swap (basic_fstream< _CharT, _Traits > &__x, basic_fstream< _CharT, _Traits > &__y)`
- `template<typename _Res >`
`void swap (promise< _Res > &__x, promise< _Res > &__y) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`void swap (__shared_ptr< _Tp, _Lp > &__a, __shared_ptr< _Tp, _Lp > &__b) noexcept`
- `template<typename _Key, typename _Val, typename _KeyOfValue, typename _Compare, typename _Alloc >`
`void swap (_Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc > &__x, _Rb_tree< _Key, _Val, _KeyOfValue, _Compare, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc >`
`void swap (map< _Key, _Tp, _Compare, _Alloc > &__x, map< _Key, _Tp, _Compare, _Alloc > &__y) noexcept(/*conditional */)`

- `template<class _Value, class _Hash, class _Pred, class _Alloc >`
`void swap (unordered_set< _Value, _Hash, _Pred, _Alloc > &__x, unordered_set< _Value, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<class _Value, class _Hash, class _Pred, class _Alloc >`
`void swap (unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Tp, typename _Alloc >`
`void swap (forward_list< _Tp, _Alloc > &__lx, forward_list< _Tp, _Alloc > &__ly) noexcept(noexcept(__lx.swap(__ly)))`
- `template<typename _Tp, _Lock_policy _Lp>`
`void swap (__weak_ptr< _Tp, _Lp > &__a, __weak_ptr< _Tp, _Lp > &__b) noexcept`
- `template<typename... _Elements>`
`void swap (tuple< _Elements... > &__x, tuple< _Elements... > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Tp, typename _Alloc >`
`void swap (vector< _Tp, _Alloc > &__x, vector< _Tp, _Alloc > &__y) noexcept(/*conditional */)`
- `template<typename _Res, typename... _ArgTypes>`
`void swap (packaged_task< _Res(_ArgTypes...)> &__x, packaged_task< _Res(_ArgTypes...)> &__y) noexcept`
- `template<class _Key, class _Tp, class _Hash, class _Pred, class _Alloc >`
`void swap (unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<class _Key, class _Tp, class _Hash, class _Pred, class _Alloc >`
`void swap (unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Tp, typename _Alloc >`
`void swap (list< _Tp, _Alloc > &__x, list< _Tp, _Alloc > &__y) noexcept(/*conditional */)`
- `template<typename _Bi_iter, typename _Alloc >`
`void swap (match_results< _Bi_iter, _Alloc > &__lhs, match_results< _Bi_iter, _Alloc > &__rhs)`
- `template<typename _Res, typename... _Args>`
`void swap (function< _Res(_Args...)> &__x, function< _Res(_Args...)> &__y) noexcept`
- `template<typename _Tp, typename _Alloc >`
`void swap (deque< _Tp, _Alloc > &__x, deque< _Tp, _Alloc > &__y) noexcept(/*conditional */)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`void swap (basic_string< _CharT, _Traits, _Alloc > &__lhs, basic_string< _CharT, _Traits, _Alloc > &__rhs) noexcept(/*conditional */)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`_ForwardIterator2 swap_ranges (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2)`
- `template<typename _Filter1, typename _Filter2 >`
`_Filter2 swap_ranges (_Filter1, _Filter1, _Filter2)`
- `template<typename _Tp >`
`complex< _Tp > tan (const complex< _Tp > &)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Tan, _ValArray, _Tp >, _Tp > tan (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< _Tan, _Expr, _Dom >, typename _Dom::value_type > tan (const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `constexpr float tan (float __x)`
- `constexpr long double tan (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::enable_if< __is_integer< _Tp >::__value, double >::__type tan (_Tp __x)`
- `template<typename _Tp >`
`complex< _Tp > tanh (const complex< _Tp > &)`

- `template<typename _Tp >`
`_Expr< _UnClos< _Tanh, _ValArray, _Tp >, _Tp > tanh (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< _Tanh, _Expr, _Dom >, typename _Dom::value_type > tanh (const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `constexpr float tanh (float __x)`
- `constexpr long double tanh (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type tanh (_Tp __x)`
- `void terminate () noexcept __attribute__((__noreturn__))`
- `template<typename _Tp >`
`void throw_with_nested (_Tp &&__t)`
- `template<typename... _Elements>`
`constexpr tuple< _Elements &... > tie (_Elements &...__args) noexcept`
- `template<typename _CharT >`
`_CharT tolower (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`_CharT toupper (_CharT __c, const locale &__loc)`
- `template<typename _Iter, typename _OIter, typename _UnaryOperation >`
`_OIter transform (_Iter, _Iter, _OIter, _UnaryOperation)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _BinaryOperation >`
`_OIter transform (_Iter1, _Iter1, _Iter2, _OIter, _BinaryOperation)`
- `template<typename _InputIterator, typename _OutputIterator, typename _UnaryOperation >`
`_OutputIterator transform (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _UnaryOperation __unary_op)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator transform (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _OutputIterator __result, _BinaryOperation __binary_op)`
- `template<typename... _Tpls, typename = typename enable_if< __and< __is_tuple_like< _Tpls>...>::__value>::__type>`
`constexpr auto tuple_cat (_Tpls &&... __tpls) -> typename __tuple_cat_result< _Tpls... >::__type`
- `bool uncaught_exception () noexcept __attribute__((__pure__))`
- `int uncaught_exceptions () noexcept __attribute__((__pure__))`
- `void undeclare_no_pointers (char *, size_t)`
- `template<class T >`
`T * undeclare_reachable (T *__p)`
- `void unexpected () __attribute__((__noreturn__))`
- `template<typename _InputIterator, typename _ForwardIterator >`
`_ForwardIterator uninitialized_copy (_InputIterator __first, _InputIterator __last, _ForwardIterator __result)`
- `template<typename _InputIterator, typename _Size, typename _ForwardIterator >`
`_ForwardIterator uninitialized_copy_n (_InputIterator __first, _Size __n, _ForwardIterator __result)`
- `template<typename _ForwardIterator, typename _Tp >`
`void uninitialized_fill (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__x)`
- `template<typename _ForwardIterator, typename _Size, typename _Tp >`
`_ForwardIterator uninitialized_fill_n (_ForwardIterator __first, _Size __n, const _Tp &__x)`
- `template<typename _Filter >`
`_Filter unique (_Filter, _Filter)`
- `template<typename _Filter, typename _BinaryPredicate >`
`_Filter unique (_Filter, _Filter, _BinaryPredicate)`
- `template<typename _ForwardIterator >`
`_ForwardIterator unique (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _BinaryPredicate >`
`_ForwardIterator unique (_ForwardIterator __first, _ForwardIterator __last, _BinaryPredicate __binary_pred)`

- `template<typename _Iter, typename _OIter >`
`_OIter unique_copy (_Iter, _Iter, _OIter)`
 - `template<typename _Iter, typename _OIter, typename _BinaryPredicate >`
`_OIter unique_copy (_Iter, _Iter, _OIter, _BinaryPredicate)`
 - `template<typename _InputIterator, typename _OutputIterator >`
`_OutputIterator unique_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result)`
 - `template<typename _InputIterator, typename _OutputIterator, typename _BinaryPredicate >`
`_OutputIterator unique_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _BinaryPredicate __binary_pred)`
 - `ios_base & unitbuf (ios_base &__base)`
 - `template<typename _Filter, typename _Tp >`
`_Filter upper_bound (_Filter, _Filter, const _Tp &)`
 - `template<typename _Filter, typename _Tp, typename _Compare >`
`_Filter upper_bound (_Filter, _Filter, const _Tp &, _Compare)`
 - `template<typename _ForwardIterator, typename _Tp >`
`_ForwardIterator upper_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __val)`
 - `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`_ForwardIterator upper_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __val, _Compare __comp)`
 - `ios_base & uppercase (ios_base &__base)`
 - `template<typename _Facet >`
`const _Facet & use_facet (const locale & __loc)`
 - `wchar_t * wcschr (wchar_t * __p, wchar_t __c)`
 - `wchar_t * wcspbrk (wchar_t * __s1, const wchar_t * __s2)`
 - `wchar_t * wcsrchr (wchar_t * __p, wchar_t __c)`
 - `wchar_t * wcsstr (wchar_t * __s1, const wchar_t * __s2)`
 - `wchar_t * wmemchr (wchar_t * __p, wchar_t __c, size_t __n)`
 - `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & ws (basic_istream< _CharT, _Traits > & __is)`
-
- `template<size_t _Nb>`
`bitset< _Nb > operator& (const bitset< _Nb > & __x, const bitset< _Nb > & __y) noexcept`
 - `template<size_t _Nb>`
`bitset< _Nb > operator| (const bitset< _Nb > & __x, const bitset< _Nb > & __y) noexcept`
 - `template<size_t _Nb>`
`bitset< _Nb > operator^ (const bitset< _Nb > & __x, const bitset< _Nb > & __y) noexcept`
-
- `template<class _CharT, class _Traits, size_t _Nb>`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, bitset< _Nb > & __x)`
 - `template<class _CharT, class _Traits, size_t _Nb>`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > & __os, const bitset< _Nb > & __x)`
-
- `template<typename _Tp >`
`complex< _Tp > operator+ (const complex< _Tp > & __x, const complex< _Tp > & __y)`
 - `template<typename _Tp >`
`complex< _Tp > operator+ (const complex< _Tp > & __x, const _Tp & __y)`
 - `template<typename _Tp >`
`complex< _Tp > operator+ (const _Tp & __x, const complex< _Tp > & __y)`

- `template<typename _Tp >`
`complex< _Tp > operator-` (const `complex< _Tp >` &__x, const `complex< _Tp >` &__y)
- `template<typename _Tp >`
`complex< _Tp > operator-` (const `complex< _Tp >` &__x, const `_Tp` &__y)
- `template<typename _Tp >`
`complex< _Tp > operator-` (const `_Tp` &__x, const `complex< _Tp >` &__y)

- `template<typename _Tp >`
`complex< _Tp > operator*` (const `complex< _Tp >` &__x, const `complex< _Tp >` &__y)
- `template<typename _Tp >`
`complex< _Tp > operator*` (const `complex< _Tp >` &__x, const `_Tp` &__y)
- `template<typename _Tp >`
`complex< _Tp > operator*` (const `_Tp` &__x, const `complex< _Tp >` &__y)

- `template<typename _Tp >`
`complex< _Tp > operator/` (const `complex< _Tp >` &__x, const `complex< _Tp >` &__y)
- `template<typename _Tp >`
`complex< _Tp > operator/` (const `complex< _Tp >` &__x, const `_Tp` &__y)
- `template<typename _Tp >`
`complex< _Tp > operator/` (const `_Tp` &__x, const `complex< _Tp >` &__y)

- `template<typename _Tp >`
`constexpr bool operator==` (const `complex< _Tp >` &__x, const `complex< _Tp >` &__y)
- `template<typename _Tp >`
`constexpr bool operator==` (const `complex< _Tp >` &__x, const `_Tp` &__y)
- `template<typename _Tp >`
`constexpr bool operator==` (const `_Tp` &__x, const `complex< _Tp >` &__y)

- `template<typename _Tp >`
`constexpr bool operator!=` (const `complex< _Tp >` &__x, const `complex< _Tp >` &__y)
- `template<typename _Tp >`
`constexpr bool operator!=` (const `complex< _Tp >` &__x, const `_Tp` &__y)
- `template<typename _Tp >`
`constexpr bool operator!=` (const `_Tp` &__x, const `complex< _Tp >` &__y)

- `template<typename _Tp >`
`reference_wrapper< _Tp > ref` (`_Tp` &__t) noexcept
- `template<typename _Tp >`
`reference_wrapper< const _Tp > cref` (const `_Tp` &__t) noexcept
- `template<typename _Tp >`
`void ref` (const `_Tp` &&)=delete
- `template<typename _Tp >`
`void cref` (const `_Tp` &&)=delete
- `template<typename _Tp >`
`reference_wrapper< _Tp > ref` (`reference_wrapper< _Tp >` __t) noexcept
- `template<typename _Tp >`
`reference_wrapper< const _Tp > cref` (`reference_wrapper< _Tp >` __t) noexcept

- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__in, _CharT &__c)`
- `template<class _Traits >`
`basic_istream< char, _Traits > & operator>> (basic_istream< char, _Traits > &__in, unsigned char &__c)`
- `template<class _Traits >`
`basic_istream< char, _Traits > & operator>> (basic_istream< char, _Traits > &__in, signed char &__c)`

- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & operator>> (basic_istream< _CharT, _Traits > &__in, _CharT *__s)`
- `template<>`
`basic_istream< char > & operator>> (basic_istream< char > &__in, char *__s)`
- `template<class _Traits >`
`basic_istream< char, _Traits > & operator>> (basic_istream< char, _Traits > &__in, unsigned char *__s)`
- `template<class _Traits >`
`basic_istream< char, _Traits > & operator>> (basic_istream< char, _Traits > &__in, signed char *__s)`

- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__out, _CharT __c)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__out, char __c)`
- `template<class _Traits >`
`basic_ostream< char, _Traits > & operator<< (basic_ostream< char, _Traits > &__out, char __c)`
- `template<class _Traits >`
`basic_ostream< char, _Traits > & operator<< (basic_ostream< char, _Traits > &__out, signed char __c)`
- `template<class _Traits >`
`basic_ostream< char, _Traits > & operator<< (basic_ostream< char, _Traits > &__out, unsigned char __c)`

- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__out, const _CharT *__s)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__out, const char *__s)`
- `template<class _Traits >`
`basic_ostream< char, _Traits > & operator<< (basic_ostream< char, _Traits > &__out, const char *__s)`
- `template<class _Traits >`
`basic_ostream< char, _Traits > & operator<< (basic_ostream< char, _Traits > &__out, const signed char *__s)`
- `template<class _Traits >`
`basic_ostream< char, _Traits > & operator<< (basic_ostream< char, _Traits > &__out, const unsigned char *__s)`

Matching, Searching, and Replacing

- `template<typename _Bi_iter, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool regex_match (_Bi_iter __s, _Bi_iter __e, match_results< _Bi_iter, _Alloc > &__m, const basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Bi_iter, typename _Ch_type, typename _Rx_traits >`
`bool regex_match (_Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Ch_type, typename _Alloc, typename _Rx_traits >`
`bool regex_match (const _Ch_type *__s, match_results< const _Ch_type *, _Alloc > &__m, const basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __f=regex_constants::match_default)`

- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool regex_match (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &__s, match_results< typename`
`basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &__m, const basic_regex< _Ch_`
`_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool regex_match (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &&, match_results< typename`
`basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &, const basic_regex< _Ch_type,`
`_Rx_traits > &, regex_constants::match_flag_type=regex_constants::match_default)=delete`
- `template<typename _Ch_type, class _Rx_traits >`
`bool regex_match (const _Ch_type *__s, const basic_regex< _Ch_type, _Rx_traits > &__re, regex_`
`constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Str_allocator, typename _Ch_type, typename _Rx_traits >`
`bool regex_match (const basic_string< _Ch_type, _Ch_traits, _Str_allocator > &__s, const basic_regex<`
`_Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Bi_iter, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool regex_search (_Bi_iter __s, _Bi_iter __e, match_results< _Bi_iter, _Alloc > &__m, const basic_regex<`
`_Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Bi_iter, typename _Ch_type, typename _Rx_traits >`
`bool regex_search (_Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type, _Rx_traits > &__re,`
`regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Ch_type, class _Alloc, class _Rx_traits >`
`bool regex_search (const _Ch_type *__s, match_results< const _Ch_type *, _Alloc > &__m, const basic_`
`_regex< _Ch_type, _Rx_traits > &__e, regex_constants::match_flag_type __f=regex_constants::match_`
`default)`
- `template<typename _Ch_type, typename _Rx_traits >`
`bool regex_search (const _Ch_type *__s, const basic_regex< _Ch_type, _Rx_traits > &__e, regex_`
`constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _String_allocator, typename _Ch_type, typename _Rx_traits >`
`bool regex_search (const basic_string< _Ch_type, _Ch_traits, _String_allocator > &__s, const basic_regex<`
`_Ch_type, _Rx_traits > &__e, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool regex_search (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &__s, match_results< typename`
`basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &__m, const basic_regex< _Ch_`
`_type, _Rx_traits > &__e, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool regex_search (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &&, match_results< typename`
`basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &, const basic_regex< _Ch_type,`
`_Rx_traits > &, regex_constants::match_flag_type=regex_constants::match_default)=delete`
- `template<typename _Out_iter, typename _Bi_iter, typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa >`
`_Out_iter regex_replace (_Out_iter __out, _Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type,`
`_Rx_traits > &__e, const basic_string< _Ch_type, _St, _Sa > &__fmt, regex_constants::match_flag_type`
`__flags=regex_constants::match_default)`
- `template<typename _Out_iter, typename _Bi_iter, typename _Rx_traits, typename _Ch_type >`
`_Out_iter regex_replace (_Out_iter __out, _Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type,`
`_Rx_traits > &__e, const _Ch_type *__fmt, regex_constants::match_flag_type __flags=regex_constants::`
`match_default)`
- `template<typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa, typename _Fst, typename _Fsa >`
`basic_string< _Ch_type, _St, _Sa > regex_replace (const basic_string< _Ch_type, _St, _Sa > &__s, const`
`basic_regex< _Ch_type, _Rx_traits > &__e, const basic_string< _Ch_type, _Fst, _Fsa > &__fmt, regex_`
`constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa >`
`basic_string< _Ch_type, _St, _Sa > regex_replace (const basic_string< _Ch_type, _St, _Sa > &__s, const`
`basic_regex< _Ch_type, _Rx_traits > &__e, const _Ch_type *__fmt, regex_constants::match_flag_type __`
`flags=regex_constants::match_default)`
- `template<typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa >`
`basic_string< _Ch_type > regex_replace (const _Ch_type *__s, const basic_regex< _Ch_type, _Rx_traits >`

- &__e, const [basic_string](#)<_Ch_type, _St, _Sa> &__fmt, [regex_constants::match_flag_type](#) __flags=[regex_constants::match_default](#))
- [template](#)<typename _Rx_traits, typename _Ch_type>
 - [basic_string](#)<_Ch_type> [regex_replace](#) (const _Ch_type * __s, const [basic_regex](#)<_Ch_type, _Rx_traits> &__e, const _Ch_type * __fmt, [regex_constants::match_flag_type](#) __flags=[regex_constants::match_default](#))

- [template](#)<typename _Tp, _Lock_policy _Lp>
 - bool [atomic_is_lock_free](#) (const [__shared_ptr](#)<_Tp, _Lp> * __p)
- [template](#)<typename _Tp>
 - bool [atomic_is_lock_free](#) (const [shared_ptr](#)<_Tp> * __p)

- [template](#)<typename _Tp>
 - [shared_ptr](#)<_Tp> [atomic_load_explicit](#) (const [shared_ptr](#)<_Tp> * __p, [memory_order](#))
- [template](#)<typename _Tp>
 - [shared_ptr](#)<_Tp> [atomic_load](#) (const [shared_ptr](#)<_Tp> * __p)
- [template](#)<typename _Tp, _Lock_policy _Lp>
 - [__shared_ptr](#)<_Tp, _Lp> [atomic_load_explicit](#) (const [__shared_ptr](#)<_Tp, _Lp> * __p, [memory_order](#))
- [template](#)<typename _Tp, _Lock_policy _Lp>
 - [__shared_ptr](#)<_Tp, _Lp> [atomic_load](#) (const [__shared_ptr](#)<_Tp, _Lp> * __p)

- [template](#)<typename _Tp>
 - void [atomic_store_explicit](#) ([shared_ptr](#)<_Tp> * __p, [shared_ptr](#)<_Tp> __r, [memory_order](#))
- [template](#)<typename _Tp>
 - void [atomic_store](#) ([shared_ptr](#)<_Tp> * __p, [shared_ptr](#)<_Tp> __r)
- [template](#)<typename _Tp, _Lock_policy _Lp>
 - void [atomic_store_explicit](#) ([__shared_ptr](#)<_Tp, _Lp> * __p, [__shared_ptr](#)<_Tp, _Lp> __r, [memory_order](#))
- [template](#)<typename _Tp, _Lock_policy _Lp>
 - void [atomic_store](#) ([__shared_ptr](#)<_Tp, _Lp> * __p, [__shared_ptr](#)<_Tp, _Lp> __r)

- [template](#)<typename _Tp>
 - [shared_ptr](#)<_Tp> [atomic_exchange_explicit](#) ([shared_ptr](#)<_Tp> * __p, [shared_ptr](#)<_Tp> __r, [memory_order](#))
- [template](#)<typename _Tp>
 - [shared_ptr](#)<_Tp> [atomic_exchange](#) ([shared_ptr](#)<_Tp> * __p, [shared_ptr](#)<_Tp> __r)
- [template](#)<typename _Tp, _Lock_policy _Lp>
 - [__shared_ptr](#)<_Tp, _Lp> [atomic_exchange_explicit](#) ([__shared_ptr](#)<_Tp, _Lp> * __p, [__shared_ptr](#)<_Tp, _Lp> __r, [memory_order](#))
- [template](#)<typename _Tp, _Lock_policy _Lp>
 - [__shared_ptr](#)<_Tp, _Lp> [atomic_exchange](#) ([__shared_ptr](#)<_Tp, _Lp> * __p, [__shared_ptr](#)<_Tp, _Lp> __r)

- [template](#)<typename _Tp>
 - bool [atomic_compare_exchange_strong_explicit](#) ([shared_ptr](#)<_Tp> * __p, [shared_ptr](#)<_Tp> * __v, [shared_ptr](#)<_Tp> __w, [memory_order](#), [memory_order](#))
- [template](#)<typename _Tp>
 - bool [atomic_compare_exchange_strong](#) ([shared_ptr](#)<_Tp> * __p, [shared_ptr](#)<_Tp> * __v, [shared_ptr](#)<_Tp> __w)
- [template](#)<typename _Tp>
 - bool [atomic_compare_exchange_weak_explicit](#) ([shared_ptr](#)<_Tp> * __p, [shared_ptr](#)<_Tp> * __v, [shared_ptr](#)<_Tp> __w, [memory_order](#) __success, [memory_order](#) __failure)

- `template<typename _Tp >`
`bool atomic_compare_exchange_weak (shared_ptr< _Tp > *__p, shared_ptr< _Tp > *__v, shared_ptr< _Tp > __w)`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool atomic_compare_exchange_strong_explicit (__shared_ptr< _Tp, _Lp > *__p, __shared_ptr< _Tp, _Lp > *__v, __shared_ptr< _Tp, _Lp > __w, memory_order, memory_order)`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool atomic_compare_exchange_strong (__shared_ptr< _Tp, _Lp > *__p, __shared_ptr< _Tp, _Lp > *__v, __shared_ptr< _Tp, _Lp > __w)`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool atomic_compare_exchange_weak_explicit (__shared_ptr< _Tp, _Lp > *__p, __shared_ptr< _Tp, _Lp > *__v, __shared_ptr< _Tp, _Lp > __w, memory_order __success, memory_order __failure)`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool atomic_compare_exchange_weak (__shared_ptr< _Tp, _Lp > *__p, __shared_ptr< _Tp, _Lp > *__v, __shared_ptr< _Tp, _Lp > __w)`

Variables

- `static ios_base::Init __ioinit`
- `template<typename _Yp, typename _Tp >`
`constexpr bool __sp_compatible_v`
- `template<typename _Tp, typename _Yp >`
`constexpr bool __sp_is_constructible_v`
- `constexpr adopt_lock_t adopt_lock`
- `constexpr allocator_arg_t allocator_arg`
- `constexpr defer_lock_t defer_lock`
- `const _Swallow_assign ignore`
- `error_code make_error_code (errc) noexcept`
- `error_condition make_error_condition (errc) noexcept`
- `const nothrow_t nothrow`
- `decltype(nullptr) typedef nullptr_t`
- `constexpr piecewise_construct_t piecewise_construct`
- `constexpr try_to_lock_t try_to_lock`

Standard Stream Objects

The `<iostream>` header declares the eight standard stream objects. For other declarations, see <http://gcc.gnu.org/onlinedocs/libstdc++/manual/io.html> and the [I/O forward declarations](#)

They are required by default to cooperate with the global C library's `FILE` streams, and to be available during program startup and termination. For more information, see the section of the manual linked to above.

- `istream cin`
 - `ostream cout`
 - `ostream cerr`
 - `ostream clog`
 - `wistream wcin`
 - `wostream wcout`
 - `wostream wcerr`
 - `wostream wclog`
-
- `__thread void * __once_callable`
 - `__thread void(* __once_call)()`

- `template<typename _Lock >`
`unique_lock< _Lock > __try_to_lock (_Lock &__l)`
- `template<typename _Lock1 , typename _Lock2 , typename... _Lock3>`
`int try_lock (_Lock1 &__l1, _Lock2 &__l2, _Lock3 &...__l3)`
- `template<typename _L1 , typename _L2 , typename... _L3>`
`void lock (_L1 &__l1, _L2 &__l2, _L3 &...__l3)`
- `void __once_proxy (void)`
- `template<typename _Callable , typename... _Args>`
`void call_once (once_flag &__once, _Callable &&__f, _Args &&...__args)`
- `using __shared_timed_mutex_base = __shared_mutex_cv`
- `template<typename _Mutex >`
`void swap (shared_lock< _Mutex > &__x, shared_lock< _Mutex > &__y) noexcept`

4.11.1 Detailed Description

ISO C++ entities toplevel namespace is std.

4.11.2 Typedef Documentation

4.11.2.1 `template<typename _Ptr , typename _Tp > using std::__ptr_rebind = typedef typename pointer_traits<_Ptr>::template rebind<_Tp>`

Convenience alias for rebinding pointers.

Definition at line 146 of file ptr_traits.h.

4.11.2.2 `template<bool _Cache> using std::__umap_traits = typedef __detail::__Hashtable_traits<_Cache, false, true>`

Base types for unordered_map.

Definition at line 39 of file unordered_map.h.

4.11.2.3 `template<bool _Cache> using std::__ummap_traits = typedef __detail::__Hashtable_traits<_Cache, false, false>`

Base types for unordered_multimap.

Definition at line 56 of file unordered_map.h.

4.11.2.4 `template<bool _Cache> using std::__umset_traits = typedef __detail::__Hashtable_traits<_Cache, true, false>`

Base types for unordered_multiset.

Definition at line 54 of file unordered_set.h.

4.11.2.5 `template<bool _Cache> using std::__uset_traits = typedef __detail::_Hashtable_traits<_Cache, true, true>`

Base types for `unordered_set`.

Definition at line 39 of file `unordered_set.h`.

4.11.2.6 `template<size_t... _Idx> using std::index_sequence = typedef integer_sequence<size_t, _Idx...>`

Alias template `index_sequence`.

Definition at line 322 of file `utility`.

4.11.2.7 `template<typename... _Types> using std::index_sequence_for = typedef make_index_sequence<sizeof...(_Types)>`

Alias template `index_sequence_for`.

Definition at line 330 of file `utility`.

4.11.2.8 `template<size_t _Num> using std::make_index_sequence = typedef make_integer_sequence<size_t, _Num>`

Alias template `make_index_sequence`.

Definition at line 326 of file `utility`.

4.11.2.9 `template<typename _Tp, _Tp _Num> using std::make_integer_sequence = typedef typename _Make_integer_sequence<_Tp, _Num>::__type`

Alias template `make_integer_sequence`.

Definition at line 318 of file `utility`.

4.11.2.10 `typedef void(* std::new_handler)()`

If you write your own error handler to be called by `new`, it must be of this type.

Definition at line 93 of file `new`.

4.11.2.11 `typedef long long std::streamoff`

Type used by `fpos`, `char_traits<char>`, and `char_traits<wchar_t>`.

In clauses 21.1.3.1 and 27.4.1 `streamoff` is described as an implementation defined type. Note: In versions of GCC up to and including GCC 3.3, `streamoff` was typedef `long`.

Definition at line 94 of file `postypes.h`.

4.11.2.12 typedef fpos<mbstate_t> std::streampos

File position for char streams.

Definition at line 228 of file postypes.h.

4.11.2.13 typedef ptrdiff_t std::streamsize

Integral type for I/O operation counts and buffer sizes.

Definition at line 98 of file postypes.h.

4.11.2.14 typedef fpos<mbstate_t> std::u16streampos

File position for char16_t streams.

Definition at line 234 of file postypes.h.

4.11.2.15 typedef fpos<mbstate_t> std::u32streampos

File position for char32_t streams.

Definition at line 236 of file postypes.h.

4.11.2.16 typedef fpos<mbstate_t> std::wstreampos

File position for wchar_t streams.

Definition at line 230 of file postypes.h.

4.11.3 Enumeration Type Documentation

4.11.3.1 anonymous enum

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation_style.html This controls some aspect of the sort routines.

Definition at line 1872 of file stl_algo.h.

4.11.3.2 enum std::float_denorm_style

Describes the denormalization for floating-point types.

These values represent the presence or absence of a variable number of exponent bits. This type is used in the std::numeric_limits class.

Enumerator

denorm_indeterminate Indeterminate at compile time whether denormalized values are allowed.

denorm_absent The type does not allow denormalized values.

denorm_present The type allows denormalized values.

Definition at line 182 of file limits.

4.11.3.3 `enum std::float_round_style`

Describes the rounding style for floating-point types.

This is used in the `std::numeric_limits` class.

Enumerator

`round_toward_zero` Intermediate.
`round_to_nearest` To zero.
`round_toward_infinity` To the nearest representable value.
`round_toward_neg_infinity` To infinity.

Definition at line 167 of file `limits`.

4.11.3.4 `enum std::io_errc` [`strong`]

I/O error code.

Definition at line 203 of file `ios_base.h`.

4.11.4 Function Documentation

4.11.4.1 `template<typename _Alloc> __allocated_ptr<_Alloc> std::__allocate_guarded (_Alloc & __a)`

Allocate space for a single object using `__a`.

Definition at line 101 of file `allocated_ptr.h`.

References `std::allocator_traits<_Alloc>::allocate()`.

4.11.4.2 `template<typename _RandomAccessIterator, typename _Compare> void std::_final_insertion_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`

This is a helper function for the sort routine.

Definition at line 1877 of file `stl_algo.h`.

References `__insertion_sort()`, and `__unguarded_insertion_sort()`.

Referenced by `__introsort_loop()`.

4.11.4.3 `template<typename _InputIterator, typename _Predicate> _InputIterator std::_find_if (_InputIterator __first, _InputIterator __last, _Predicate __pred, input_iterator_tag)` [`inline`]

This is an overload used by find algos for the Input Iterator case.

Definition at line 101 of file `stl_algo.h`.

Referenced by `__find_if()`, `__find_if_not()`, `__find_if_not_n()`, `__search_n_aux()`, `find()`, `find_if()`, `is_permutation()`, `minmax_element()`, and `partition_copy()`.

4.11.4.4 `template<typename _RandomAccessIterator, typename _Predicate> _RandomAccessIterator std::__find_if (`
`_RandomAccessIterator __first, _RandomAccessIterator __last, _Predicate __pred, random_access_iterator_tag)`

This is an overload used by find algos for the RAI case.

Definition at line 112 of file `stl_algo.h`.

References `__find_if()`, and `__iterator_category()`.

4.11.4.5 `template<typename _InputIterator, typename _Predicate> _InputIterator std::__find_if_not (_InputIterator __first,`
`_InputIterator __last, _Predicate __pred) [inline]`

Provided for `stable_partition` to use.

Definition at line 168 of file `stl_algo.h`.

References `__find_if()`, and `__iterator_category()`.

Referenced by `__stable_partition_adaptive()`, and `find_if_not()`.

4.11.4.6 `template<typename _InputIterator, typename _Predicate, typename _Distance> _InputIterator std::__find_if_not_n (`
`_InputIterator __first, _Distance & __len, _Predicate __pred)`

Like `find_if_not()`, but uses and updates a count of the remaining range length instead of comparing against an end iterator.

Definition at line 181 of file `stl_algo.h`.

References `__find_if()`.

Referenced by `__stable_partition_adaptive()`.

4.11.4.7 `template<typename _EuclideanRingElement> _EuclideanRingElement std::__gcd (_EuclideanRingElement __m,`
`_EuclideanRingElement __n)`

This is a helper function for the rotate algorithm specialized on RAIs. It returns the greatest common divisor of two integer values.

Definition at line 1229 of file `stl_algo.h`.

4.11.4.8 `template<typename _RandomAccessIterator, typename _Compare> void std::__heap_select (_RandomAccessIterator`
`__first, _RandomAccessIterator __middle, _RandomAccessIterator __last, _Compare __comp)`

This is a helper function for the sort routines.

Definition at line 1665 of file `stl_algo.h`.

Referenced by `__introsort_loop()`, and `__unguarded_partition_pivot()`.

4.11.4.9 `template<typename _RandomAccessIterator, typename _Compare> void std::__inplace_stable_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`

This is a helper function for the stable sorting routines.

Definition at line 2758 of file `stl_algo.h`.

References `__insertion_sort()`, and `__merge_without_buffer()`.

Referenced by `merge()`.

4.11.4.10 `template<typename _RandomAccessIterator, typename _Compare> void std::__insertion_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`

This is a helper function for the sort routine.

Definition at line 1837 of file `stl_algo.h`.

References `__unguarded_linear_insert()`.

Referenced by `__final_insertion_sort()`, `__inplace_stable_sort()`, `__introsort_loop()`, and `__move_merge()`.

4.11.4.11 `template<typename _RandomAccessIterator, typename _Size, typename _Compare> void std::__introsort_loop (_RandomAccessIterator __first, _RandomAccessIterator __last, _Size __depth_limit, _Compare __comp)`

This is a helper function for the sort routine.

Definition at line 1937 of file `stl_algo.h`.

References `__final_insertion_sort()`, `__heap_select()`, `__insertion_sort()`, `__lg()`, `__unguarded_partition_pivot()`, and `iter_swap()`.

4.11.4.12 `template<typename _Callable, typename... _Args> result_of<_Callable&&(_Args&&...)>::type std::__invoke (_Callable && __fn, _Args &&... __args) [inline]`

Invoke a callable object.

Definition at line 245 of file `functional`.

4.11.4.13 `constexpr int std::__lg (int __n) [inline]`

This is a helper function for the sort routines and for `random.tcc`.

Definition at line 1000 of file `stl_algobase.h`.

Referenced by `__introsort_loop()`, `nth_element()`, `std::independent_bits_engine<_RandomNumberEngine, __w, _U, IntType>::operator()()`, and `std::linear_congruential_engine<_UIntType, __a, __c, __m>::seed()`.

4.11.4.14 `template<typename _BidirectionalIterator , typename _Distance , typename _Pointer , typename _Compare > void
std::__merge_adaptive (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last,
_Distance __len1, _Distance __len2, _Pointer __buffer, _Distance __buffer_size, _Compare __comp)`

This is a helper function for the merge routines.

Definition at line 2412 of file `stl_algo.h`.

References `__move_merge_adaptive()`, `__move_merge_adaptive_backward()`, `__rotate_adaptive()`, `advance()`, and `distance()`.

Referenced by `__merge_without_buffer()`, and `__move_merge()`.

4.11.4.15 `template<typename _BidirectionalIterator , typename _Distance , typename _Compare > void
std::__merge_without_buffer (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last,
_Distance __len1, _Distance __len2, _Compare __comp)`

This is a helper function for the merge routines.

Definition at line 2473 of file `stl_algo.h`.

References `__merge_adaptive()`, `advance()`, `distance()`, `iter_swap()`, and `std::_V2::rotate()`.

Referenced by `__inplace_stable_sort()`.

4.11.4.16 `template<typename _Iterator , typename _Compare > void std::__move_median_to_first (_Iterator __result, _Iterator
__a, _Iterator __b, _Iterator __c, _Compare __comp)`

Swaps the median value of `*__a`, `*__b` and `*__c` under `__comp` to `*__result`.

Definition at line 78 of file `stl_algo.h`.

References `iter_swap()`.

Referenced by `__unguarded_partition_pivot()`.

4.11.4.17 `template<typename _InputIterator , typename _OutputIterator , typename _Compare > _OutputIterator
std::__move_merge (_InputIterator __first1, _InputIterator __last1, _InputIterator __first2, _InputIterator __last2,
_OutputIterator __result, _Compare __comp)`

This is a helper function for the `__merge_sort_loop` routines.

Definition at line 2636 of file `stl_algo.h`.

References `__insertion_sort()`, `__merge_adaptive()`, and `min()`.

4.11.4.18 `template<typename _InputIterator1 , typename _InputIterator2 , typename _OutputIterator , typename _Compare > void
std::__move_merge_adaptive (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2
__last2, _OutputIterator __result, _Compare __comp)`

This is a helper function for the `__merge_adaptive` routines.

Definition at line 2301 of file `stl_algo.h`.

Referenced by `__merge_adaptive()`.

4.11.4.19 `template<typename _BidirectionalIterator1, typename _BidirectionalIterator2, typename _BidirectionalIterator3, typename _Compare > void std::__move_merge_adaptive_backward (_BidirectionalIterator1 __first1, _BidirectionalIterator1 __last1, _BidirectionalIterator2 __first2, _BidirectionalIterator2 __last2, _BidirectionalIterator3 __result, _Compare __comp)`

This is a helper function for the `__merge_adaptive` routines.

Definition at line 2327 of file `stl_algo.h`.

Referenced by `__merge_adaptive()`.

4.11.4.20 `void std::__once_proxy (void)`

Generic `try_lock`.

Parameters

<code>__l1</code>	Meets Mutex requirements (<code>try_lock()</code> may throw).
<code>__l2</code>	Meets Mutex requirements (<code>try_lock()</code> may throw).
<code>__l3</code>	Meets Mutex requirements (<code>try_lock()</code> may throw).

Returns

Returns -1 if all `try_lock()` calls return true. Otherwise returns a 0-based index corresponding to the argument that returned false.

Postcondition

Either all arguments are locked, or none will be.

Sequentially calls `try_lock()` on each argument.

4.11.4.21 `template<typename _ForwardIterator, typename _Predicate > _ForwardIterator std::__partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred, forward_iterator_tag)`

This is a helper function...

Definition at line 1485 of file `stl_algo.h`.

References `iter_swap()`.

Referenced by `partition()`.

```
4.11.4.22 template<typename _BidirectionalIterator , typename _Predicate > _BidirectionalIterator std::__partition (
    _BidirectionalIterator __first, _BidirectionalIterator __last, _Predicate __pred, bidirectional_iterator_tag )
```

This is a helper function...

Definition at line 1510 of file stl_algo.h.

References iter_swap().

```
4.11.4.23 template<typename _BidirectionalIterator > void std::__reverse ( _BidirectionalIterator __first, _BidirectionalIterator
    __last, bidirectional_iterator_tag )
```

This is an uglified reverse(_BidirectionalIterator, _BidirectionalIterator) overloaded for bidirectional iterators.

Definition at line 1129 of file stl_algo.h.

References iter_swap().

Referenced by std::_V2::__rotate(), includes(), next_permutation(), and reverse().

```
4.11.4.24 template<typename _RandomAccessIterator > void std::__reverse ( _RandomAccessIterator __first,
    _RandomAccessIterator __last, random_access_iterator_tag )
```

This is an uglified reverse(_BidirectionalIterator, _BidirectionalIterator) overloaded for random access iterators.

Definition at line 1149 of file stl_algo.h.

References iter_swap().

```
4.11.4.25 template<typename _BidirectionalIterator1 , typename _BidirectionalIterator2 , typename _Distance >
    _BidirectionalIterator1 std::__rotate_adaptive ( _BidirectionalIterator1 __first, _BidirectionalIterator1 __middle,
    _BidirectionalIterator1 __last, _Distance __len1, _Distance __len2, _BidirectionalIterator2 __buffer, _Distance
    __buffer_size )
```

This is a helper function for the merge routines.

Definition at line 2370 of file stl_algo.h.

References advance(), distance(), and std::_V2::rotate().

Referenced by __merge_adaptive().

```
4.11.4.26 template<typename _ForwardIterator , typename _Integer , typename _UnaryPredicate > _ForwardIterator
    std::__search_n_aux ( _ForwardIterator __first, _ForwardIterator __last, _Integer __count, _UnaryPredicate
    __unary_pred, std::forward_iterator_tag )
```

This is an helper function for search_n overloaded for forward iterators.

Definition at line 257 of file stl_algo.h.

References __find_if().

Referenced by __search_n_aux().


```
4.11.4.27 template<typename _RandomAccessIter, typename _Integer, typename _UnaryPredicate > _RandomAccessIter
std::__search_n_aux( _RandomAccessIter __first, _RandomAccessIter __last, _Integer __count, _UnaryPredicate
__unary_pred, std::random_access_iterator_tag )
```

This is an helper function for search_n overloaded for random access iterators.

Definition at line 289 of file stl_algo.h.

References __find_if(), __iterator_category(), __search_n_aux(), advance(), and distance().

```
4.11.4.28 template<typename _ForwardIterator, typename _Pointer, typename _Predicate, typename _Distance > _ForwardIterator
std::__stable_partition_adaptive( _ForwardIterator __first, _ForwardIterator __last, _Predicate __pred, _Distance __len,
 Pointer __buffer, _Distance __buffer_size )
```

This is a helper function... Requires __first != __last and !__pred(__first) and __len == distance(__first, __last).

!__pred(__first) allows us to guarantee that we don't move-assign an element onto itself.

Definition at line 1546 of file stl_algo.h.

References __find_if_not(), __find_if_not_n(), advance(), std::Temporary_buffer< _ForwardIterator, _Tp >::begin(), distance(), std::Temporary_buffer< _ForwardIterator, _Tp >::requested_size(), std::V2::rotate(), and std::__Temporary_buffer< _ForwardIterator, _Tp >::size().

```
4.11.4.29 template<typename _Lock > unique_lock<_Lock> std::try_to_lock( _Lock & __l ) [inline]
```

Generic try_lock.

Parameters

\leftrightarrow __l1	Meets Mutex requirements (try_lock() may throw).
\leftrightarrow __l2	Meets Mutex requirements (try_lock() may throw).
\leftrightarrow __l3	Meets Mutex requirements (try_lock() may throw).

Returns

Returns -1 if all try_lock() calls return true. Otherwise returns a 0-based index corresponding to the argument that returned false.

Postcondition

Either all arguments are locked, or none will be.

Sequentially calls try_lock() on each argument.

Definition at line 467 of file mutex.

4.11.4.30 `template<typename _RandomAccessIterator, typename _Compare> void std::__unguarded_insertion_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp) [inline]`

This is a helper function for the sort routine.

Definition at line 1860 of file `stl_algo.h`.

References `__unguarded_linear_insert()`.

Referenced by `__final_insertion_sort()`.

4.11.4.31 `template<typename _RandomAccessIterator, typename _Compare> void std::__unguarded_linear_insert (_RandomAccessIterator __last, _Compare __comp)`

This is a helper function for the sort routine.

Definition at line 1818 of file `stl_algo.h`.

Referenced by `__insertion_sort()`, and `__unguarded_insertion_sort()`.

4.11.4.32 `template<typename _RandomAccessIterator, typename _Compare> _RandomAccessIterator std::__unguarded_partition (_RandomAccessIterator __first, _RandomAccessIterator __last, _RandomAccessIterator __pivot, _Compare __comp)`

This is a helper function...

Definition at line 1893 of file `stl_algo.h`.

References `iter_swap()`.

Referenced by `__unguarded_partition_pivot()`.

4.11.4.33 `template<typename _RandomAccessIterator, typename _Compare> _RandomAccessIterator std::__unguarded_partition_pivot (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp) [inline]`

This is a helper function...

Definition at line 1914 of file `stl_algo.h`.

References `__heap_select()`, `__move_median_to_first()`, and `__unguarded_partition()`.

Referenced by `__introsort_loop()`.

4.11.4.34 `template<typename _ForwardIterator, typename _OutputIterator, typename _BinaryPredicate> _OutputIterator std::__unique_copy (_ForwardIterator __first, _ForwardIterator __last, _OutputIterator __result, _BinaryPredicate __binary_pred, forward_iterator_tag, output_iterator_tag)`

This is an uglified `unique_copy(_InputIterator, _InputIterator, _OutputIterator, _BinaryPredicate)` overloaded for forward iterators and output iterator as result.

Definition at line 1046 of file `stl_algo.h`.

Referenced by `unique_copy()`.

```
4.11.4.35  template<typename _InputIterator , typename _OutputIterator , typename _BinaryPredicate > _OutputIterator
std::__unique_copy ( _InputIterator __first, _InputIterator __last, _OutputIterator __result, _BinaryPredicate
__binary_pred, input_iterator_tag , output_iterator_tag )
```

This is an uglified `unique_copy(_InputIterator, _InputIterator, _OutputIterator, _BinaryPredicate)` overloaded for input iterators and output iterator as result.

Definition at line 1075 of file `stl_algo.h`.

```
4.11.4.36  template<typename _InputIterator , typename _ForwardIterator , typename _BinaryPredicate > _ForwardIterator
std::__unique_copy ( _InputIterator __first, _InputIterator __last, _ForwardIterator __result, _BinaryPredicate
__binary_pred, input_iterator_tag , forward_iterator_tag )
```

This is an uglified `unique_copy(_InputIterator, _InputIterator, _OutputIterator, _BinaryPredicate)` overloaded for input iterators and forward iterator as result.

Definition at line 1107 of file `stl_algo.h`.

```
4.11.4.37  template<typename _T1 , typename... _Args> void std::_Construct ( _T1 * __p, _Args &&... __args ) [inline]
```

Constructs an object in existing memory by invoking an allocated object's constructor with an initializer.

Definition at line 74 of file `stl_construct.h`.

Referenced by `std::Temporary_buffer< _ForwardIterator, _Tp >::end()`, `uninitialized_copy()`, `uninitialized_fill()`, and `uninitialized_fill_n()`.

```
4.11.4.38  template<typename _Tp > void std::_Destroy ( _Tp * __pointer ) [inline]
```

Destroy the object pointed to by a pointer type.

Definition at line 92 of file `stl_construct.h`.

References `__addressof()`.

Referenced by `_Destroy()`, `std::vector< sub_match< _Bi_iter >, allocator< sub_match< _Bi_iter > > >::M_← allocate_and_copy()`, `std::deque< _Tp, _Alloc >::M_fill_initialize()`, `std::deque< _Tp, _Alloc >::M_pop_front_← aux()`, `std::deque< _Tp, _Alloc >::M_range_initialize()`, `std::vector< _Tp, _Alloc >::emplace()`, `std::Temporary_← buffer< _ForwardIterator, _Tp >::end()`, `std::vector< _Tp, _Alloc >::operator=()`, `std::vector< _Tp, _Alloc >::reserve()`, `uninitialized_copy()`, `uninitialized_fill()`, `uninitialized_fill_n()`, and `std::vector< sub_match< _Bi_iter >, allocator< sub_← _match< _Bi_iter > > >::~~vector()`.

```
4.11.4.39  template<typename _ForwardIterator > void std::_Destroy ( _ForwardIterator __first, _ForwardIterator __last )
[inline]
```

Destroy a range of objects. If the `value_type` of the object has a trivial destructor, the compiler should optimize all of this away, otherwise the objects' destructors must be invoked.

Definition at line 122 of file `stl_construct.h`.

4.11.4.40 `template<typename _ForwardIterator, typename _Allocator> void std::_Destroy (_ForwardIterator __first, _ForwardIterator __last, _Allocator & __alloc)`

Destroy a range of objects using the supplied allocator. For nondefault allocators we do not optimize away invocation of `destroy()` even if `_Tp` has a trivial destructor.

Definition at line 138 of file `stl_construct.h`.

References `__addressof()`, and `_Destroy()`.

4.11.4.41 `template<typename _InputIterator, typename _Tp> _Tp std::accumulate (_InputIterator __first, _InputIterator __last, _Tp __init) [inline]`

Accumulate values in a range.

Accumulates the values in the range `[first,last)` using `operator+()`. The initial value is `init`. The values are processed in order.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__init</code>	Starting value to add other values to.

Returns

The final sum.

Definition at line 120 of file `stl_numeric.h`.

Referenced by `__gnu_parallel::__parallel_partial_sum_linear()`, and `operator>>()`.

4.11.4.42 `template<typename _InputIterator, typename _Tp, typename _BinaryOperation> _Tp std::accumulate (_InputIterator __first, _InputIterator __last, _Tp __init, _BinaryOperation __binary_op) [inline]`

Accumulate values in a range with operation.

Accumulates the values in the range `[first,last)` using the function object `__binary_op`. The initial value is `__init`. The values are processed in order.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__init</code>	Starting value to add other values to.
<code>__binary_op</code>	Function object to accumulate with.

Returns

The final sum.

Definition at line 146 of file `stl_numeric.h`.

4.11.4.43 `template<typename _Tp> std::complex<_Tp> std::acos (const std::complex<_Tp> & __z) [inline]`

`acos(__z)` [8.1.2].

Definition at line 1617 of file `complex`.

4.11.4.44 `template<typename _Tp> std::complex<_Tp> std::acosh (const std::complex<_Tp> & __z) [inline]`

`acosh(__z)` [8.1.5].

Definition at line 1733 of file `complex`.

4.11.4.45 `template<typename _InputIterator, typename _OutputIterator> _OutputIterator std::adjacent_difference (_InputIterator __first, _InputIterator __last, _OutputIterator __result)`

Return differences between adjacent values.

Computes the difference between adjacent values in the range `[first,last)` using operator-() and writes the result to `__result`.

Parameters

<code>__first</code>	Start of input range.
<code>__last</code>	End of input range.
<code>__result</code>	Output sums.

Returns

Iterator pointing just beyond the values written to result.

`_GLIBCXX_RESOLVE_LIB_DEFECTS` DR 539. `partial_sum` and `adjacent_difference` should mention requirements

Definition at line 317 of file `stl_numeric.h`.

4.11.4.46 `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation> _OutputIterator std::adjacent_difference (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _BinaryOperation __binary_op)`

Return differences between adjacent values.

Computes the difference between adjacent values in the range `[__first,__last)` using the function object `__binary_op` and writes the result to `__result`.

Parameters

<code>__first</code>	Start of input range.
<code>__last</code>	End of input range.
<code>__result</code>	Output sum.
<code>__binary_op</code>	Function object.

Returns

Iterator pointing just beyond the values written to result.

`_GLIBCXX_RESOLVE_LIB_DEFECTS` DR 539. `partial_sum` and `adjacent_difference` should mention requirements

Definition at line 360 of file `stl_numeric.h`.

4.11.4.47 `template<typename _InputIterator, typename _Distance> void std::advance (_InputIterator & __i, _Distance __n)`
`[inline]`

A generalization of pointer arithmetic.

Parameters

<code>__i</code>	An input iterator.
<code>__n</code>	The <i>delta</i> by which to change <code>__i</code> .

Returns

Nothing.

This increments `i` by `n`. For bidirectional and random access iterators, `__n` may be negative, in which case `__i` is decremented.

For random access iterators, this uses their `+` and `-` operations and are constant time. For other iterator classes they are linear time.

Definition at line 194 of file `stl_iterator_base_funcs.h`.

References `__iterator_category()`.

Referenced by `__merge_adaptive()`, `__merge_without_buffer()`, `__rotate_adaptive()`, `__search_n_aux()`, `__stable_partition_adaptive()`, `std::deque< _Tp, _Alloc >::M_pop_front_aux()`, `std::deque< _Tp, _Alloc >::M_range_initialize()`, `std::deque< _StateSeqT >::clear()`, `std::vector< _Tp, _Alloc >::emplace()`, `std::list< _Tp, _Alloc >::erase()`, `fill_n()`, `__gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _Clterator, Iterator, _Alloc >::get_child()`, `__gnu_pbds::detail::pat_trie_base::_Node_iter< Node, Leaf, Head, Inode, _Clterator, Iterator, _Alloc >::get_child()`, `lower_bound()`, `minmax_element()`, `std::vector< _Tp, _Alloc >::operator=()`, `partition_point()`, and `upper_bound()`.

4.11.4.48 `void* std::align (size_t __align, size_t __size, void *& __ptr, size_t & __space) [inline], [noexcept]`

Fit aligned storage in buffer.

[ptr.align]

This function tries to fit `__size` bytes of storage with alignment `__align` into the buffer `__ptr` of size `__space` bytes. If such a buffer fits then `__ptr` is changed to point to the first byte of the aligned storage and `__space` is reduced by the bytes used for alignment.

Parameters

<code>__align</code>	A fundamental or extended alignment value.
<code>__size</code>	Size of the aligned storage required.
<code>__ptr</code>	Pointer to a buffer of <code>__space</code> bytes.
<code>__space</code>	Size of the buffer pointed to by <code>__ptr</code> .

Returns

the updated pointer if the aligned storage fits, otherwise `nullptr`.

Definition at line 115 of file `memory`.

4.11.4.49 `template<typename _Tp> __gnu_cxx::__promote<_Tp>::__type std::arg (_Tp __x) [inline]`

Additional overloads [8.1.9].

Definition at line 1831 of file `complex`.

4.11.4.50 `template<typename _Tp> std::complex<_Tp> std::asin (const std::complex<_Tp> & __z) [inline]`

`asin(__z)` [8.1.3].

Definition at line 1653 of file `complex`.

4.11.4.51 `template<typename _Tp> std::complex<_Tp> std::asinh (const std::complex<_Tp> & __z) [inline]`

`asinh(__z)` [8.1.6].

Definition at line 1772 of file `complex`.

4.11.4.52 `template<typename _Tp> std::complex<_Tp> std::atan (const std::complex<_Tp> & __z) [inline]`

`atan(__z)` [8.1.4].

Definition at line 1697 of file `complex`.

4.11.4.53 `template<typename _Tp> std::complex<_Tp> std::atanh (const std::complex<_Tp> & __z) [inline]`

`atanh(__z)` [8.1.7].

Definition at line 1816 of file `complex`.

4.11.4.54 `template<typename _Container> auto std::begin (_Container & __cont)-> decltype(__cont.begin()) [inline]`

Return an iterator pointing to the first element of the container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

Definition at line 48 of file range_access.h.

4.11.4.55 `template<typename _Container> auto std::begin (const _Container & __cont) -> decltype(__cont.begin())`
`[inline]`

Return an iterator pointing to the first element of the const container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

Definition at line 58 of file range_access.h.

4.11.4.56 `template<typename _Tp, size_t _Nm> _GLIBCXX14_CONSTEXPR _Tp* std::begin (_Tp(&) __arr[_Nm])` `[inline]`

Return an iterator pointing to the first element of the array.

Parameters

<code>__arr</code>	Array.
--------------------	--------

Definition at line 87 of file range_access.h.

4.11.4.57 `template<class _Tp> constexpr const _Tp* std::begin (initializer_list<_Tp> __ils)` `[noexcept]`

Return an iterator pointing to the first element of the initializer_list.

Parameters

<code>__ils</code>	Initializer list.
--------------------	-------------------

Definition at line 89 of file initializer_list.

Referenced by `__gnu_profile::__report()`, `std::deque<_StateSeqT>::__M_reserve_map_at_front()`, `std::match_results<_Bi_iter>::begin()`, `cbegin()`, `std::match_results<_Bi_iter>::cbegin()`, `std::deque<_StateSeqT>::clear()`, `std::vector<sub_match<_Bi_iter>, allocator<sub_match<_Bi_iter>>>::crend()`, `std::list<__inp, __rebind_inp>::crend()`, `std::vector<_Tp, _Alloc>::emplace()`, `std::vector<sub_match<_Bi_iter>, allocator<sub_match<_Bi_iter>>>::empty()`, `end()`, `std::vector<sub_match<_Bi_iter>, allocator<sub_match<_Bi_iter>>>::erase()`, `std::list<_Tp, _Alloc>::erase()`, `std::vector<sub_match<_Bi_iter>, allocator<sub_match<_Bi_iter>>>::front()`, `std::list<__inp, __rebind_inp>::front()`, `std::deque<_StateSeqT>::front()`, `__gnu_pbds::detail::pat_trie_base::__Node_iter<Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc>::get_child()`, `std::vector<_Tp, _Alloc>::insert()`,

std::vector< sub_match< _Bi_iter >, allocator< sub_match< _Bi_iter > > >::insert(), std::deque< _Tp, _Alloc >::insert(), std::deque< _StateSeqT >::insert(), std::list< __inp, __rebind_inp >::list(), std::list< _Tp, _Alloc >::merge(), __gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >::node_end(), std::vector< _Tp, _Alloc >::operator=(), std::list< _Tp, _Alloc >::operator=(), std::list< __inp, __rebind_inp >::pop_front(), std::list< __inp, __rebind_inp >::push_front(), std::list< _Tp, _Alloc >::remove(), std::list< _Tp, _Alloc >::remove_if(), std::vector< sub_match< _Bi_iter >, allocator< sub_match< _Bi_iter > > >::rend(), std::list< __inp, __rebind_inp >::rend(), std::forward_list< _Tp, _Alloc >::reverse(), std::list< _Tp, _Alloc >::sort(), std::forward_list< _Tp, _Alloc >::unique(), std::list< _Tp, _Alloc >::unique(), and std::deque< _StateSeqT >::~~deque().

4.11.4.58 ios_base& std::boolalpha (ios_base & __base) [inline]

Calls base.setf(ios_base::boolalpha).

Definition at line 878 of file ios_base.h.

References std::ios_base::boolalpha, and std::ios_base::setf().

4.11.4.59 template<typename _Callable, typename... _Args> void std::call_once (once_flag & __once, _Callable && __f, _Args &&... __args)

call_once

Definition at line 597 of file mutex.

4.11.4.60 template<typename _Container > constexpr auto std::cbegin (const _Container & __cont) -> decltype(std::begin(__cont)) [inline], [noexcept]

Return an iterator pointing to the first element of the const container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

Definition at line 116 of file range_access.h.

References begin().

Referenced by std::vector< _Tp, _Alloc >::emplace(), std::vector< sub_match< _Bi_iter >, allocator< sub_match< _Bi_iter > > >::erase(), std::vector< _Tp, _Alloc >::insert(), std::vector< sub_match< _Bi_iter >, allocator< sub_match< _Bi_iter > > >::insert(), and std::deque< _StateSeqT >::insert().

4.11.4.61 template<typename _Container > constexpr auto std::cend (const _Container & __cont) -> decltype(std::end(__cont)) [inline], [noexcept]

Return an iterator pointing to one past the last element of the const container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

Definition at line 127 of file range_access.h.

References end().

4.11.4.62 `template<typename _Tp, typename _Tp1, _Lock_policy _Lp> __shared_ptr<_Tp, _Lp> std::const_pointer_cast (const __shared_ptr<_Tp1, _Lp> &__r) [inline], [noexcept]`

const_pointer_cast

Definition at line 1324 of file shared_ptr_base.h.

4.11.4.63 `template<typename _Container > auto std::crbegin (const _Container & __cont)-> decltype(std::rbegin(__cont)) [inline]`

Return a reverse iterator pointing to the last element of the const container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

Definition at line 218 of file range_access.h.

References rbegin().

4.11.4.64 `template<typename _Tp > reference_wrapper<const _Tp> std::cref (const _Tp &__t) [inline], [noexcept]`

Denotes a const reference should be taken to a variable.

Definition at line 479 of file functional.

4.11.4.65 `template<typename _Tp > void std::cref (const _Tp &&) [delete]`

Denotes a reference should be taken to a variable.

4.11.4.66 `template<typename _Tp > reference_wrapper<const _Tp> std::cref (reference_wrapper<_Tp> __t) [inline], [noexcept]`

Partial specialization.

Definition at line 497 of file functional.

4.11.4.67 `template<typename _Container > auto std::crend (const _Container & __cont)-> decltype(std::rend(__cont)) [inline]`

Return a reverse iterator pointing one past the first element of the const container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

Definition at line 228 of file `range_access.h`.

References `rend()`.

4.11.4.68 `ios_base& std::dec (ios_base & __base) [inline]`

Calls `base.setf(ios_base::dec, ios_base::basefield)`.

Definition at line 1016 of file `ios_base.h`.

References `std::ios_base::basefield`, `std::ios_base::dec`, and `std::ios_base::setf()`.

Referenced by `std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >::discard()`, `std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::operator()()`, `std::discard_block_engine< _RandomNumberEngine, __p, __r >::operator()()`, `std::shuffle_order_engine< _RandomNumberEngine, __k >::operator()()`, `std::normal_distribution< _RealType >::operator()()`, `std::gamma_distribution< _RealType >::operator()()`, `std::binomial_distribution< _IntType >::operator()()`, `operator>>()`, and `std::linear_congruential_engine< _UIntType, __a, __c, __m >::seed()`.

4.11.4.69 `ios_base& std::defaultfloat (ios_base & __base) [inline]`

Calls `base.unsetf(ios_base::floatfield)`

Definition at line 1069 of file `ios_base.h`.

References `std::ios_base::floatfield`, and `std::ios_base::unsetf()`.

4.11.4.70 `template<typename _InputIterator > iterator_traits<_InputIterator>::difference_type std::distance (_InputIterator __first, _InputIterator __last) [inline]`

A generalization of pointer arithmetic.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Returns

The distance between them.

Returns `n` such that `__first + n == __last`. This requires that `__last` must be reachable from `__first`. Note that `n` may be negative.

For random access iterators, this uses their + and – operations and are constant time. For other iterator classes they are linear time.

Definition at line 135 of file `stl_iterator_base_funcs.h`.

References `__iterator_category()`.

Referenced by `__merge_adaptive()`, `__merge_without_buffer()`, `__rotate_adaptive()`, `__search_n_aux()`, `__stable_partition_adaptive()`, `std::vector< sub_match< _Bi_iter >, allocator< sub_match< _Bi_iter > >::__M_allocate_and_copy()`, `std::deque< _Tp, _Alloc >::__M_pop_front_aux()`, `std::deque< _Tp, _Alloc >::__M_range_initialize()`, `std::deque< _StateSeqT >::clear()`, `std::vector< _Tp, _Alloc >::emplace()`, `equal()`, `fill_n()`, `__gnu_pbds::detail::pat_trie_base::Node_iter< Node, Leaf, Head, Inode, _Cliterator, Iterator, _Alloc >::get_child()`, `is_heap_until()`, `is_permutation()`, `std::sub_match< _Bi_iter >::length()`, `lower_bound()`, `minmax_element()`, `__gnu_parallel::multiseq_partition()`, `__gnu_parallel::multiseq_selection()`, `__gnu_pbds::detail::pat_trie_base::Node_citer< Node, Leaf, Head, Inode, _Cliterator, Iterator, _Alloc >::num_children()`, `std::vector< _Tp, _Alloc >::operator=()`, `partition_point()`, `std::match_results< _Bi_iter >::position()`, and `upper_bound()`.

4.11.471 `template<typename _Tp, typename _Tp1, _Lock_policy _Lp> __shared_ptr<_Tp, _Lp> std::dynamic_pointer_cast (const __shared_ptr<_Tp1, _Lp> &_r) [inline], [noexcept]`

`dynamic_pointer_cast`

Definition at line 1334 of file `shared_ptr_base.h`.

References `lock()`.

4.11.472 `template<typename _Container> auto std::end (_Container &__cont)-> decltype(__cont.end()) [inline]`

Return an iterator pointing to one past the last element of the container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

Definition at line 68 of file `range_access.h`.

4.11.473 `template<typename _Container> auto std::end (const _Container &__cont)-> decltype(__cont.end()) [inline]`

Return an iterator pointing to one past the last element of the const container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

Definition at line 78 of file `range_access.h`.

4.11.474 `template<typename _Tp, size_t _Nm> _GLIBCXX14_CONSTEXPR _Tp* std::end (_Tp(&)__arr[_Nm]) [inline]`

Return an iterator pointing to one past the last element of the array.

Parameters

<code>__arr</code>	Array.
--------------------	--------

Definition at line 97 of file `range_access.h`.

References `begin()`, and `end()`.

4.11.4.75 `template<class _Tp> constexpr const _Tp* std::end (initializer_list<_Tp> __ils) [noexcept]`

Return an iterator pointing to one past the last element of the `initializer_list`.

Parameters

<code>__ils</code>	Initializer list.
--------------------	-------------------

Definition at line 99 of file `initializer_list`.

Referenced by `__gnu_profile::__report()`, `std::vector< sub_match< _Bi_iter >, allocator< sub_match< _Bi_iter > > >::back()`, `std::list< __inp, __rebind_inp >::back()`, `std::deque< _StateSeqT >::back()`, `std::match_results< _Bi_iter >::end()`, `std::deque< _StateSeqT >::clear()`, `std::vector< sub_match< _Bi_iter >, allocator< sub_match< _Bi_iter > > >::crbegin()`, `std::list< __inp, __rebind_inp >::crbegin()`, `std::vector< _Tp, _Alloc >::emplace()`, `std::vector< sub_match< _Bi_iter >, allocator< sub_match< _Bi_iter > > >::empty()`, `std::match_results< _Bi_iter >::end()`, `std::list< _Tp, _Alloc >::erase()`, `__gnu_pbds::detail::pat_trie_base::_Node_iter< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::get_child()`, `std::vector< _Tp, _Alloc >::insert()`, `std::deque< _Tp, _Alloc >::insert()`, `std::list< _Tp, _Alloc >::merge()`, `__gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >::node_end()`, `std::vector< _Tp, _Alloc >::operator=()`, `std::list< _Tp, _Alloc >::operator=()`, `std::vector< sub_match< _Bi_iter >, allocator< sub_match< _Bi_iter > > >::push_back()`, `std::list< __inp, __rebind_inp >::push_back()`, `std::vector< sub_match< _Bi_iter >, allocator< sub_match< _Bi_iter > > >::rbegin()`, `std::list< __inp, __rebind_inp >::rbegin()`, `std::list< _Tp, _Alloc >::remove()`, `std::list< _Tp, _Alloc >::remove_if()`, `std::vector< sub_match< _Bi_iter >, allocator< sub_match< _Bi_iter > > >::resize()`, `std::list< _Tp, _Alloc >::resize()`, `std::forward_list< _Tp, _Alloc >::resize()`, `std::forward_list< _Tp, _Alloc >::reverse()`, `std::forward_list< _Tp, _Alloc >::unique()`, `std::list< _Tp, _Alloc >::unique()`, and `std::deque< _StateSeqT >::~deque()`.

4.11.4.76 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>& std::endl (basic_ostream<_CharT, _Traits> & __os) [inline]`

Write a newline and flush the stream.

This manipulator is often mistakenly used when a simple newline is desired, leading to poor buffering performance. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html#io.streambuf.buffering> for more on this subject.

Definition at line 590 of file `ostream`.

Referenced by `operator<<()`.

4.11.4.77 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>& std::ends (basic_ostream<_CharT, _Traits> & __os) [inline]`

Write a null character into the output sequence.

Null character is `CharT()` by definition. For `CharT` of `char`, this correctly writes the ASCII NUL character string terminator.

Definition at line 602 of file `ostream`.

Referenced by operator<<().

4.11.4.78 `template<typename _Tp, typename _Up = _Tp> _Tp std::exchange (_Tp & __obj, _Up && __new_val) [inline]`

Assign `__new_val` to `__obj` and return its previous value.

Definition at line 254 of file `utility`.

4.11.4.79 `template<typename _Tp> _Tp std::fabs (const std::complex<_Tp> & __z) [inline]`

`fabs(__z)` [8.1.8].

Definition at line 1825 of file `complex`.

4.11.4.80 `ios_base& std::fixed (ios_base & __base) [inline]`

Calls `base.setf(ios_base::fixed, ios_base::floatfield)`.

Definition at line 1041 of file `ios_base.h`.

References `std::ios_base::fixed`, `std::ios_base::floatfield`, and `std::ios_base::setf()`.

Referenced by `std::mersenne_twister_engine<_UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f>::discard()`, `std::subtract_with_carry_engine<_UIntType, __w, __s, __r>::operator()()`, `std::discard_block_engine<_RandomNumberEngine, __p, __r>::operator()()`, `std::shuffle_order_engine<_RandomNumberEngine, __k>::operator()()`, and `std::linear_congruential_engine<_UIntType, __a, __c, __m>::seed()`.

4.11.4.81 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>& std::flush (basic_ostream<_CharT, _Traits> & __os) [inline]`

Flushes the output stream.

This manipulator simply calls the stream's `flush()` member function.

Definition at line 612 of file `ostream`.

4.11.4.82 `template<typename _MoneyT> _Get_money<_MoneyT> std::get_money (_MoneyT & __mon, bool __intl = false) [inline]`

Extended manipulator for extracting money.

Parameters

<code>__mon</code>	Either long double or a specialization of <code>basic_string</code> .
<code>__intl</code>	A bool indicating whether international format is to be used.

Sent to a stream object, this manipulator extracts `__mon`.

Definition at line 259 of file `iomanip`.

4.11.4.83 `new_handler` `std::get_new_handler ()` `[noexcept]`

Return the current new handler.

4.11.4.84 `template<typename _Tp> pair<_Tp*, ptrdiff_t> std::get_temporary_buffer (ptrdiff_t __len)` `[noexcept]`

Allocates a temporary buffer.

Parameters

<code>__len</code>	The number of objects of type <code>Tp</code> .
--------------------	---

Returns

See full description.

Reinventing the wheel, but this time with prettier spokes!

This function tries to obtain storage for `__len` adjacent `Tp` objects. The objects themselves are not constructed, of course. A `pair<>` is returned containing *the buffer's address and capacity (in the units of `sizeof(_Tp)`)*, or a pair of 0 values if no storage can be obtained. Note that the capacity obtained may be less than that requested if the memory is unavailable; you should compare `len` with the `.second` return value.

Provides the nothrow exception guarantee.

Definition at line 85 of file `stl_tempbuf.h`.

Referenced by `std::_Temporary_buffer<_ForwardIterator, _Tp>::_Temporary_buffer()`.

4.11.4.85 `template<typename _CharT> _Get_time<_CharT> std::get_time (std::tm * __tmb, const _CharT * __fmt)` `[inline]`

Extended manipulator for extracting time.

This manipulator uses `time_get::get` to extract time. `[ext.manip]`

Parameters

<code>__tmb</code>	struct to extract the time data to.
<code>__fmt</code>	format string.

Definition at line 413 of file iomanip.

```
4.11.4.86 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename >
class _Base> basic_istream< _CharT, _Traits > & std::getline ( basic_istream< _CharT, _Traits > & __is,
__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base > & __str, _CharT __delim )
```

Read a line from stream into a string.

Parameters

<code>__is</code>	Input stream.
<code>__str</code>	Buffer to store into.
<code>__delim</code>	Character marking end of line.

Returns

Reference to the input stream.

Stores characters from `__is` into `__str` until `__delim` is found, the end of the stream is encountered, or `str.max_size()` is reached. If `is.width()` is non-zero, that is the limit on the number of characters stored into `__str`. Any previous contents of `__str` are erased. If `delim` was encountered, it is extracted but not stored into `__str`.

Definition at line 627 of file vstring.tcc.

References `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::append()`, `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::erase()`, `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::max_size()`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

```
4.11.4.87 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename >
class _Base> basic_istream<_CharT, _Traits>& std::getline ( basic_istream< _CharT, _Traits > & __is,
__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base > & __str ) [inline]
```

Read a line from stream into a string.

Parameters

<code>__is</code>	Input stream.
<code>__str</code>	Buffer to store into.

Returns

Reference to the input stream.

Stores characters from `is` into `__str` until '`'` is found, the end of the stream is encountered, or `str.max_size()` is reached. If `is.width()` is non-zero, that is the limit on the number of characters stored into `__str`. Any previous contents of `__str` are erased. If end of line was encountered, it is extracted but not stored into `__str`.

Definition at line 2676 of file vstring.h.

References `std::basic_ios<_CharT, _Traits>::widen()`.

Referenced by operator<<().

4.11.4.88 `template<typename _CharT, typename _Traits, typename _Alloc> basic_istream<_CharT, _Traits> & std::getline (basic_istream<_CharT, _Traits> & __is, basic_string<_CharT, _Traits, _Alloc> & __str, _CharT __delim)`

Read a line from stream into a string.

Parameters

<code>__is</code>	Input stream.
<code>__str</code>	Buffer to store into.
<code>__delim</code>	Character marking end of line.

Returns

Reference to the input stream.

Stores characters from `__is` into `__str` until `__delim` is found, the end of the stream is encountered, or `str.max_size()` is reached. Any previous contents of `__str` are erased. If `__delim` is encountered, it is extracted but not stored into `__str`.

Definition at line 1509 of file basic_string.tcc.

References `std::basic_string<_CharT, _Traits, _Alloc>::erase()`, `std::basic_string<_CharT, _Traits, _Alloc>::max_size()`, `operator>>()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

Referenced by `__gnu_profile::__report()`, `getline()`, and operator<<().

4.11.4.89 `template<typename _CharT, typename _Traits, typename _Alloc> basic_istream<_CharT, _Traits> & std::getline (basic_istream<_CharT, _Traits> & __is, basic_string<_CharT, _Traits, _Alloc> & __str) [inline]`

Read a line from stream into a string.

Parameters

<code>__is</code>	Input stream.
<code>__str</code>	Buffer to store into.

Returns

Reference to the input stream.

Stores characters from `is` into `__str` until ' ' is found, the end of the stream is encountered, or `str.max_size()` is reached. Any previous contents of `__str` are erased. If end of line is encountered, it is extracted but not stored into `__str`.

Definition at line 5365 of file basic_string.h.

References `getline()`, and `std::basic_ios<_CharT, _Traits>::widen()`.

```
4.11.4.90 template<typename _CharT, typename _Traits, typename _Alloc> basic_istream<_CharT, _Traits>& std::getline (
    basic_istream<_CharT, _Traits> && __is, basic_string<_CharT, _Traits, _Alloc> & __str, _CharT __delim )
    [inline]
```

Read a line from an rvalue stream into a string.

Definition at line 5373 of file basic_string.h.

References `getline()`.

```
4.11.4.91 template<typename _CharT, typename _Traits, typename _Alloc> basic_istream<_CharT, _Traits>& std::getline (
    basic_istream<_CharT, _Traits> && __is, basic_string<_CharT, _Traits, _Alloc> & __str ) [inline]
```

Read a line from an rvalue stream into a string.

Definition at line 5380 of file basic_string.h.

References `std::basic_string<_CharT, _Traits, _Alloc>::c_str()`, and `getline()`.

```
4.11.4.92 ios_base& std::hex ( ios_base & __base ) [inline]
```

Calls `base.setf(ios_base::hex, ios_base::basefield)`.

Definition at line 1024 of file ios_base.h.

References `std::ios_base::basefield`, `std::ios_base::hex`, and `std::ios_base::setf()`.

Referenced by `std::regex_traits<_Ch_type>::value()`.

```
4.11.4.93 ios_base& std::hexfloat ( ios_base & __base ) [inline]
```

Calls `base.setf(ios_base::fixed|ios_basescientific, ios_base::floatfield)`

Definition at line 1061 of file ios_base.h.

References `std::ios_base::fixed`, `std::ios_base::floatfield`, `std::ios_base::scientific`, and `std::ios_base::setf()`.

```
4.11.4.94 template<typename _InputIterator1, typename _InputIterator2, typename _Tp> _Tp std::inner_product ( _InputIterator1
    __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Tp __init ) [inline]
```

Compute inner product of two ranges.

Starting with an initial value of `__init`, multiplies successive elements from the two ranges and adds each product into the accumulated value using `operator+()`. The values in the ranges are processed in order.

Parameters

<code>__first1</code>	Start of range 1.
<code>__last1</code>	End of range 1.
<code>__first2</code>	Start of range 2.
<code>__init</code>	Starting value to add other values to.

Returns

The final inner product.

Definition at line 174 of file `stl_numeric.h`.

```
4.11.4.95 template<typename _InputIterator1, typename _InputIterator2, typename _Tp, typename _BinaryOperation1, typename
    _BinaryOperation2 > _Tp std::inner_product ( _InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2,
    _Tp __init, _BinaryOperation1 __binary_op1, _BinaryOperation2 __binary_op2 ) [inline]
```

Compute inner product of two ranges.

Starting with an initial value of `__init`, applies `__binary_op2` to successive elements from the two ranges and accumulates each result into the accumulated value using `__binary_op1`. The values in the ranges are processed in order.

Parameters

<code>__first1</code>	Start of range 1.
<code>__last1</code>	End of range 1.
<code>__first2</code>	Start of range 2.
<code>__init</code>	Starting value to add other values to.
<code>__binary_op1</code>	Function object to accumulate with.
<code>__binary_op2</code>	Function object to apply to pairs of input values.

Returns

The final inner product.

Definition at line 206 of file `stl_numeric.h`.

```
4.11.4.96 ios_base& std::internal ( ios_base & __base ) [inline]
```

Calls `base.setf(ios_base::internal, ios_base::adjustfield)`.

Definition at line 991 of file `ios_base.h`.

References `std::ios_base::adjustfield`, and `std::ios_base::internal`.

4.11.4.97 `template<typename _ForwardIterator, typename _Tp> void std::iota (_ForwardIterator __first, _ForwardIterator __last, _Tp __value)`

Create a range of sequentially increasing values.

For each element in the range [first,last) assigns `value` and increments `value` as if by `++value`.

Parameters

<code>__first</code>	Start of range.
<code>__last</code>	End of range.
<code>__value</code>	Starting value.

Returns

Nothing.

Definition at line 82 of file `stl_numeric.h`.

4.11.4.98 `template<typename _CharT> bool std::isalnum (_CharT __c, const locale & __loc) [inline]`

Convenience interface to `ctype.is(ctype_base::alnum, __c)`.

Definition at line 2619 of file `locale_facets.h`.

References `std::__ctype_abstract_base< _CharT >::is()`.

4.11.4.99 `template<typename _CharT> bool std::isalpha (_CharT __c, const locale & __loc) [inline]`

Convenience interface to `ctype.is(ctype_base::alpha, __c)`.

Definition at line 2595 of file `locale_facets.h`.

References `std::__ctype_abstract_base< _CharT >::is()`.

4.11.4.100 `template<typename _CharT> bool std::isblank (_CharT __c, const locale & __loc) [inline]`

Convenience interface to `ctype.is(ctype_base::blank, __c)`.

Definition at line 2632 of file `locale_facets.h`.

References `std::__ctype_abstract_base< _CharT >::is()`.

4.11.4.101 `template<typename _CharT> bool std::iscntrl (_CharT __c, const locale & __loc) [inline]`

Convenience interface to `ctype.is(ctype_base::cntrl, __c)`.

Definition at line 2577 of file `locale_facets.h`.

References `std::__ctype_abstract_base< _CharT >::is()`.

4.11.4.102 `template<typename _CharT> bool std::isdigit (_CharT __c, const locale & __loc) [inline]`

Convenience interface to `ctype.is(ctype_base::digit, __c)`.

Definition at line 2601 of file `locale_facets.h`.

References `std::__ctype_abstract_base< _CharT >::is()`.

4.11.4.103 `template<typename _CharT> bool std::isgraph (_CharT __c, const locale & __loc) [inline]`

Convenience interface to `ctype.is(ctype_base::graph, __c)`.

Definition at line 2625 of file `locale_facets.h`.

References `std::__ctype_abstract_base< _CharT >::is()`.

4.11.4.104 `template<typename _CharT> bool std::islower (_CharT __c, const locale & __loc) [inline]`

Convenience interface to `ctype.is(ctype_base::lower, __c)`.

Definition at line 2589 of file `locale_facets.h`.

References `std::__ctype_abstract_base< _CharT >::is()`.

4.11.4.105 `template<typename _CharT> bool std::isprint (_CharT __c, const locale & __loc) [inline]`

Convenience interface to `ctype.is(ctype_base::print, __c)`.

Definition at line 2571 of file `locale_facets.h`.

References `std::__ctype_abstract_base< _CharT >::is()`.

4.11.4.106 `template<typename _CharT> bool std::ispunct (_CharT __c, const locale & __loc) [inline]`

Convenience interface to `ctype.is(ctype_base::punct, __c)`.

Definition at line 2607 of file `locale_facets.h`.

References `std::__ctype_abstract_base< _CharT >::is()`.

4.11.4.107 `template<typename _CharT> bool std::isspace (_CharT __c, const locale & __loc) [inline]`

Convenience interface to `ctype.is(ctype_base::space, __c)`.

Definition at line 2565 of file `locale_facets.h`.

References `std::__ctype_abstract_base< _CharT >::is()`.

4.11.4.108 `template<typename _CharT> bool std::isupper (_CharT __c, const locale & __loc) [inline]`

Convenience interface to `ctype.is(ctype_base::upper, __c)`.

Definition at line 2583 of file `locale_facets.h`.

References `std::__ctype_abstract_base< _CharT >::is()`.

4.11.4.109 `template<typename _CharT> bool std::isxdigit (_CharT __c, const locale & __loc) [inline]`

Convenience interface to `ctype.is(ctype_base::xdigit, __c)`.

Definition at line 2613 of file `locale_facets.h`.

References `std::__ctype_abstract_base< _CharT >::is()`.

4.11.4.110 `ios_base& std::left (ios_base & __base) [inline]`

Calls `base.setf(ios_base::left, ios_base::adjustfield)`.

Definition at line 999 of file `ios_base.h`.

References `std::ios_base::adjustfield`, `std::ios_base::left`, and `std::ios_base::setf()`.

Referenced by `std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >::discard()`, `std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::operator()`, `std::discard_block_engine< _RandomNumberEngine, __p, __r >::operator()`, `std::shuffle_order_engine< _RandomNumberEngine, __k >::operator()`, `std::normal_distribution< _RealType >::operator()`, `std::gamma_distribution< _RealType >::operator()`, `std::binomial_distribution< _IntType >::operator()`, `std::negative_binomial_distribution< _IntType >::operator()`, `std::poisson_distribution< _IntType >::operator()`, `operator<<()`, `operator>>()`, and `std::linear_congruential_engine< _UIntType, __a, __c, __m >::seed()`.

4.11.4.111 `template<typename _L1, typename _L2, typename... _L3> void std::lock (_L1 & __l1, _L2 & __l2, _L3 &... __l3)`

Generic lock.

Parameters

<code>__l1</code>	Meets Mutex requirements (try_lock() may throw).
<code>__l2</code>	Meets Mutex requirements (try_lock() may throw).
<code>__l3</code>	Meets Mutex requirements (try_lock() may throw).

Exceptions

<i>An</i>	exception thrown by an argument's <code>lock()</code> or <code>try_lock()</code> member.
-----------	--

Postcondition

All arguments are locked.

All arguments are locked via a sequence of calls to `lock()`, `try_lock()` and `unlock()`. If the call exits via an exception any locks that were obtained will be released.

Definition at line 540 of file `mutex`.

Referenced by `dynamic_pointer_cast()`.

4.11.4.112 ios_base& std::noboolalpha (ios_base & __base) [inline]

Calls base.unsetf(ios_base::boolalpha).

Definition at line 886 of file ios_base.h.

References std::ios_base::boolalpha, and std::ios_base::unsetf().

4.11.4.113 ios_base& std::noshowbase (ios_base & __base) [inline]

Calls base.unsetf(ios_base::showbase).

Definition at line 902 of file ios_base.h.

References std::ios_base::showbase, and std::ios_base::unsetf().

4.11.4.114 ios_base& std::noshowpoint (ios_base & __base) [inline]

Calls base.unsetf(ios_base::showpoint).

Definition at line 918 of file ios_base.h.

References std::ios_base::showpoint, and std::ios_base::unsetf().

4.11.4.115 ios_base& std::noshowpos (ios_base & __base) [inline]

Calls base.unsetf(ios_base::showpos).

Definition at line 934 of file ios_base.h.

References std::ios_base::showpos, and std::ios_base::unsetf().

4.11.4.116 ios_base& std::noskipws (ios_base & __base) [inline]

Calls base.unsetf(ios_base::skipws).

Definition at line 950 of file ios_base.h.

References std::ios_base::skipws, and std::ios_base::unsetf().

4.11.4.117 ios_base& std::nounitbuf (ios_base & __base) [inline]

Calls base.unsetf(ios_base::unitbuf).

Definition at line 982 of file ios_base.h.

References std::ios_base::unitbuf, and std::ios_base::unsetf().

4.11.4.118 `ios_base& std::nouppercase (ios_base & __base) [inline]`

Calls `base.unsetf(ios_base::uppercase)`.

Definition at line 966 of file `ios_base.h`.

References `std::ios_base::unsetf()`, and `std::ios_base::uppercase`.

4.11.4.119 `ios_base& std::oct (ios_base & __base) [inline]`

Calls `base.setf(ios_base::oct, ios_base::basefield)`.

Definition at line 1032 of file `ios_base.h`.

References `std::ios_base::basefield`, `std::ios_base::oct`, and `std::ios_base::setf()`.

Referenced by `std::regex_traits<_Ch_type>::value()`.

4.11.4.120 `template<typename _Tp> bool std::operator!= (const _Fwd_list_iterator<_Tp> & __x, const
_Fwd_list_const_iterator<_Tp> & __y) [inline], [noexcept]`

Forward list iterator inequality comparison.

Definition at line 266 of file `forward_list.h`.

4.11.4.121 `template<typename _Tp, typename _Seq> bool std::operator!= (const stack<_Tp, _Seq> & __x, const stack<
_Tp, _Seq> & __y) [inline]`

Based on `operator==`.

Definition at line 299 of file `stl_stack.h`.

4.11.4.122 `template<typename _Tp, typename _Seq> bool std::operator!= (const queue<_Tp, _Seq> & __x, const queue<
_Tp, _Seq> & __y) [inline]`

Based on `operator==`.

Definition at line 322 of file `stl_queue.h`.

4.11.4.123 `template<typename _Key, typename _Compare, typename _Alloc> bool std::operator!= (const multiset<_Key,
_Compare, _Alloc> & __x, const multiset<_Key, _Compare, _Alloc> & __y) [inline]`

Returns `!(x == y)`.

Definition at line 847 of file `stl_multiset.h`.

4.11.4.124 `template<typename _Key, typename _Compare, typename _Alloc> bool std::operator!= (const set<_Key,
_Compare, _Alloc> & __x, const set<_Key, _Compare, _Alloc> & __y) [inline]`

Returns `!(x == y)`.

Definition at line 864 of file `stl_set.h`.

```
4.11.4.125 template<typename _Key , typename _Tp , typename _Compare , typename _Alloc > bool std::operator!= ( const
    multimap< _Key, _Tp, _Compare, _Alloc > &__x, const multimap< _Key, _Tp, _Compare, _Alloc > &__y )
    [inline]
```

Based on operator==.

Definition at line 996 of file stl_multimap.h.

```
4.11.4.126 template<typename _Key , typename _Tp , typename _Compare , typename _Alloc > bool std::operator!= ( const
    map< _Key, _Tp, _Compare, _Alloc > &__x, const map< _Key, _Tp, _Compare, _Alloc > &__y ) [inline]
```

Based on operator==.

Definition at line 1331 of file stl_map.h.

```
4.11.4.127 template<typename _Tp , typename _Alloc > bool std::operator!= ( const forward_list< _Tp, _Alloc > &__x, const
    forward_list< _Tp, _Alloc > &__y ) [inline]
```

Based on operator==.

Definition at line 1384 of file forward_list.h.

```
4.11.4.128 template<typename _Tp , typename _Alloc > bool std::operator!= ( const vector< _Tp, _Alloc > &__x, const
    vector< _Tp, _Alloc > &__y ) [inline]
```

Based on operator==.

Definition at line 1533 of file stl_vector.h.

```
4.11.4.129 template<typename _Tp , typename _Alloc > bool std::operator!= ( const list< _Tp, _Alloc > &__x, const list< _Tp,
    _Alloc > &__y ) [inline]
```

Based on operator==.

Definition at line 1894 of file stl_list.h.

```
4.11.4.130 template<typename _Res , typename... _Args> bool std::operator!= ( const function< _Res(_Args...)> &__f, nullptr_t
    ) [inline], [noexcept]
```

Compares a polymorphic function object wrapper against 0 (the NULL pointer).

Returns

`false` if the wrapper has no target, `true` otherwise

This function will not throw an exception.

Definition at line 2211 of file functional.

4.11.4.131 `template<typename _Res, typename... _Args> bool std::operator!= (nullptr_t, const function< _Res(_Args...)> & __f) [inline], [noexcept]`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.

Definition at line 2217 of file functional.

4.11.4.132 `template<typename _Tp, typename _Alloc > bool std::operator!= (const deque< _Tp, _Alloc > & __x, const deque< _Tp, _Alloc > & __y) [inline]`

Based on operator==.

Definition at line 2246 of file stl_deque.h.

4.11.4.133 `template<typename _CharT, typename _Traits, typename _Alloc > bool std::operator!= (const basic_string< _CharT, _Traits, _Alloc > & __lhs, const basic_string< _CharT, _Traits, _Alloc > & __rhs) [inline], [noexcept]`

Test difference of two strings.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs.compare(__rhs) != 0`. False otherwise.

Definition at line 5097 of file basic_string.h.

4.11.4.134 `template<typename _CharT, typename _Traits, typename _Alloc > bool std::operator!= (const _CharT * __lhs, const basic_string< _CharT, _Traits, _Alloc > & __rhs) [inline]`

Test difference of C string and string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__rhs.compare(__lhs) != 0`. False otherwise.

Definition at line 5110 of file basic_string.h.

4.11.4.135 `template<typename _CharT, typename _Traits, typename _Alloc> bool std::operator!= (const basic_string<_CharT, _Traits, _Alloc> & __lhs, const _CharT* __rhs) [inline]`

Test difference of string and C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs.compare(__rhs) != 0`. False otherwise.

Definition at line 5122 of file `basic_string.h`.

4.11.4.136 `template<size_t _Nb> bitset<_Nb> std::operator& (const bitset<_Nb> & __x, const bitset<_Nb> & __y) [inline], [noexcept]`

Global bitwise operations on bitsets.

Parameters

<code>__x</code>	A bitset.
<code>__y</code>	A bitset of the same size as <code>__x</code> .

Returns

A new bitset.

These should be self-explanatory.

Definition at line 1425 of file `bitset`.

4.11.4.137 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string<_CharT, _Traits, _Alloc> std::operator+ (const basic_string<_CharT, _Traits, _Alloc> & __lhs, const basic_string<_CharT, _Traits, _Alloc> & __rhs)`

Concatenate two strings.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Last string.

Returns

New string with value of `__lhs` followed by `__rhs`.

Definition at line 4929 of file `basic_string.h`.

References `operator+()`.

4.11.4.138 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string<_CharT, _Traits, _Alloc>
std::operator+ (const _CharT* __lhs, const basic_string<_CharT, _Traits, _Alloc> &__rhs)`

Concatenate C string and string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Last string.

Returns

New string with value of `__lhs` followed by `__rhs`.

Definition at line 1147 of file `basic_string.tcc`.

References `std::basic_string<_CharT, _Traits, _Alloc>::size()`.

4.11.4.139 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string<_CharT, _Traits, _Alloc>
std::operator+ (_CharT __lhs, const basic_string<_CharT, _Traits, _Alloc> &__rhs)`

Concatenate character and string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Last string.

Returns

New string with `__lhs` followed by `__rhs`.

Definition at line 1163 of file `basic_string.tcc`.

References `std::basic_string<_CharT, _Traits, _Alloc>::find()`, and `std::basic_string<_CharT, _Traits, _Alloc>::size()`.

4.11.4.140 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string<_CharT, _Traits, _Alloc>
std::operator+ (const basic_string<_CharT, _Traits, _Alloc> &__lhs, const _CharT* __rhs) [inline]`

Concatenate string and C string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Last string.

Returns

New string with `__lhs` followed by `__rhs`.

Definition at line 4966 of file `basic_string.h`.

```
4.11.4.141 template<typename _CharT, typename _Traits, typename _Alloc> basic_string<_CharT, _Traits, _Alloc>
std::operator+ ( const basic_string<_CharT, _Traits, _Alloc> & __lhs, _CharT __rhs ) [inline]
```

Concatenate string and character.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Last string.

Returns

New string with `__lhs` followed by `__rhs`.

Definition at line 4982 of file `basic_string.h`.

References `operator+()`.

```
4.11.4.142 template<typename _Tp, typename _Seq> bool std::operator< ( const stack<_Tp, _Seq> & __x, const stack<
_Tp, _Seq> & __y ) [inline]
```

Stack ordering relation.

Parameters

<code>__x</code>	A stack.
<code>__y</code>	A stack of the same type as <code>x</code> .

Returns

True iff `x` is lexicographically less than `__y`.

This is an total ordering relation. Complexity and semantics depend on the underlying sequence type, but the expected

rules are: this relation is linear in the size of the sequences, the elements must be comparable with `<`, and `std::lexicographical_compare()` is usually used to make the determination.

Definition at line 293 of file `stl_stack.h`.

```
4.11.4.143 template<typename _Tp, typename _Seq > bool std::operator< ( const queue< _Tp, _Seq > & __x, const queue<
    _Tp, _Seq > & __y ) [inline]
```

Queue ordering relation.

Parameters

<code>__x</code>	A queue.
<code>__y</code>	A queue of the same type as <code>x</code> .

Returns

True iff `__x` is lexicographically less than `__y`.

This is an total ordering relation. Complexity and semantics depend on the underlying sequence type, but the expected rules are: this relation is linear in the size of the sequences, the elements must be comparable with `<`, and `std::lexicographical_compare()` is usually used to make the determination.

Definition at line 316 of file `stl_queue.h`.

References `std::queue< _Tp, _Sequence >::c`.

```
4.11.4.144 template<typename _Key, typename _Compare, typename _Alloc > bool std::operator< ( const multiset< _Key,
    _Compare, _Alloc > & __x, const multiset< _Key, _Compare, _Alloc > & __y ) [inline]
```

Multiset ordering relation.

Parameters

<code>__x</code>	A multiset.
<code>__y</code>	A multiset of the same type as <code>__x</code> .

Returns

True iff `__x` is lexicographically less than `__y`.

This is a total ordering relation. It is linear in the size of the sets. The elements must be comparable with `<`.

See `std::lexicographical_compare()` for how the determination is made.

Definition at line 840 of file `stl_multiset.h`.

4.11.4.145 `template<typename _Key , typename _Compare , typename _Alloc > bool std::operator< (const set< _Key, _Compare, _Alloc > & __x, const set< _Key, _Compare, _Alloc > & __y) [inline]`

Set ordering relation.

Parameters

<code>__x</code>	A set.
<code>__y</code>	A set of the same type as <code>x</code> .

Returns

True iff `__x` is lexicographically less than `__y`.

This is a total ordering relation. It is linear in the size of the sets. The elements must be comparable with `<`.

See `std::lexicographical_compare()` for how the determination is made.

Definition at line 857 of file `stl_set.h`.

4.11.4.146 `template<typename _Key , typename _Tp , typename _Compare , typename _Alloc > bool std::operator< (const multimap< _Key, _Tp, _Compare, _Alloc > & __x, const multimap< _Key, _Tp, _Compare, _Alloc > & __y) [inline]`

Multimap ordering relation.

Parameters

<code>__x</code>	A multimap.
<code>__y</code>	A multimap of the same type as <code>__x</code> .

Returns

True iff `x` is lexicographically less than `y`.

This is a total ordering relation. It is linear in the size of the multimaps. The elements must be comparable with `<`.

See `std::lexicographical_compare()` for how the determination is made.

Definition at line 989 of file `stl_multimap.h`.

4.11.4.147 `template<typename _Key , typename _Tp , typename _Compare , typename _Alloc > bool std::operator< (const map< _Key, _Tp, _Compare, _Alloc > & __x, const map< _Key, _Tp, _Compare, _Alloc > & __y) [inline]`

Map ordering relation.

Parameters

$_x$	A map.
$_y$	A map of the same type as x .

Returns

True iff x is lexicographically less than y .

This is a total ordering relation. It is linear in the size of the maps. The elements must be comparable with $<$.

See `std::lexicographical_compare()` for how the determination is made.

Definition at line 1324 of file `std_map.h`.

4.11.4.148 `template<typename _Tp, typename _Alloc> bool std::operator< (const forward_list< _Tp, _Alloc> & __lx, const forward_list< _Tp, _Alloc> & __ly) [inline]`

Forward list ordering relation.

Parameters

$_lx$	A <code>forward_list</code> .
$_ly$	A <code>forward_list</code> of the same type as $_lx$.

Returns

True iff $_lx$ is lexicographically less than $_ly$.

This is a total ordering relation. It is linear in the number of elements of the forward lists. The elements must be comparable with $<$.

See `std::lexicographical_compare()` for how the determination is made.

Definition at line 1376 of file `forward_list.h`.

References `std::forward_list< _Tp, _Alloc>::cbegin()`, `std::forward_list< _Tp, _Alloc>::cend()`, and `lexicographical_compare()`.

4.11.4.149 `template<typename _Tp, typename _Alloc> bool std::operator< (const vector< _Tp, _Alloc> & __x, const vector< _Tp, _Alloc> & __y) [inline]`

Vector ordering relation.

Parameters

$_x$	A vector.
$_y$	A vector of the same type as $_x$.

Returns

True iff $_x$ is lexicographically less than $_y$.

This is a total ordering relation. It is linear in the size of the vectors. The elements must be comparable with $<$.

See `std::lexicographical_compare()` for how the determination is made.

Definition at line 1526 of file `std_vector.h`.

References `lexicographical_compare()`.

4.11.4.150 `template<typename _Tp, typename _Alloc> bool std::operator< (const list< _Tp, _Alloc > & _x, const list< _Tp, _Alloc > & _y) [inline]`

List ordering relation.

Parameters

$_x$	A list.
$_y$	A list of the same type as $_x$.

Returns

True iff $_x$ is lexicographically less than $_y$.

This is a total ordering relation. It is linear in the size of the lists. The elements must be comparable with $<$.

See `std::lexicographical_compare()` for how the determination is made.

Definition at line 1887 of file `std_list.h`.

References `lexicographical_compare()`.

4.11.4.151 `template<typename _Tp, typename _Alloc> bool std::operator< (const deque< _Tp, _Alloc > & _x, const deque< _Tp, _Alloc > & _y) [inline]`

Deque ordering relation.

Parameters

<code>__x</code>	A deque.
<code>__y</code>	A deque of the same type as <code>__x</code> .

Returns

True iff `x` is lexicographically less than `__y`.

This is a total ordering relation. It is linear in the size of the deques. The elements must be comparable with `<`.

See `std::lexicographical_compare()` for how the determination is made.

Definition at line 2238 of file `std_deque.h`.

References `lexicographical_compare()`.

```
4.11.4.152 template<typename _CharT, typename _Traits, typename _Alloc > bool std::operator< ( const basic_string<
    _CharT, _Traits, _Alloc > & __lhs, const basic_string< _CharT, _Traits, _Alloc > & __rhs ) [inline],
    [noexcept]
```

Test if string precedes string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs` precedes `__rhs`. False otherwise.

Definition at line 5135 of file `basic_string.h`.

References `std::basic_string< _CharT, _Traits, _Alloc >::compare()`.

```
4.11.4.153 template<typename _CharT, typename _Traits, typename _Alloc > bool std::operator< ( const basic_string<
    _CharT, _Traits, _Alloc > & __lhs, const _CharT * __rhs ) [inline]
```

Test if string precedes C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs` precedes `__rhs`. False otherwise.

Definition at line 5148 of file `basic_string.h`.

References `std::basic_string<_CharT, _Traits, _Alloc>::compare()`.

4.11.4.154 `template<typename _CharT, typename _Traits, typename _Alloc> bool std::operator< (const _CharT * __lhs, const basic_string<_CharT, _Traits, _Alloc> & __rhs) [inline]`

Test if C string precedes string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__lhs` precedes `__rhs`. False otherwise.

Definition at line 5160 of file `basic_string.h`.

4.11.4.155 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>& std::operator<< (basic_ostream<_CharT, _Traits> & __out, _CharT __c) [inline]`

Character inserters.

Parameters

<code>__out</code>	An output stream.
<code>__c</code>	A character.

Returns

`out`

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts a single character and any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

If `__c` is of type `char` and the character type of the stream is not `char`, the character is widened before insertion.

Definition at line 497 of file `ostream`.

4.11.4.156 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>& std::operator<< (basic_ostream<_CharT, _Traits> & __out, char __c) [inline]`

Character inserters.

Parameters

<code>__out</code>	An output stream.
<code>__c</code>	A character.

Returns

`out`

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts a single character and any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

If `__c` is of type `char` and the character type of the stream is not `char`, the character is widened before insertion.

Definition at line 502 of file `ostream`.

```
4.11.4.157 template<class _Traits> basic_ostream<char, _Traits>& std::operator<< ( basic_ostream< char, _Traits> &
    __out, char __c ) [inline]
```

Character inserters.

Parameters

<code>__out</code>	An output stream.
<code>__c</code>	A character.

Returns

`out`

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts a single character and any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

If `__c` is of type `char` and the character type of the stream is not `char`, the character is widened before insertion.

Definition at line 508 of file `ostream`.

```
4.11.4.158 template<class _Traits> basic_ostream<char, _Traits>& std::operator<< ( basic_ostream< char, _Traits> &
    __out, signed char __c ) [inline]
```

Character inserters.

Parameters

<code>__out</code>	An output stream.
<code>__c</code>	A character.

Returns

out

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts a single character and any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

If `__c` is of type `char` and the character type of the stream is not `char`, the character is widened before insertion.

Definition at line 514 of file `ostream`.

```
4.11.4.159 template<class _Traits> basic_ostream<char, _Traits>& std::operator<< ( basic_ostream< char, _Traits> &
    __out, unsigned char __c ) [inline]
```

Character inserters.

Parameters

<code>__out</code>	An output stream.
<code>__c</code>	A character.

Returns

out

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts a single character and any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

If `__c` is of type `char` and the character type of the stream is not `char`, the character is widened before insertion.

Definition at line 519 of file `ostream`.

```
4.11.4.160 template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>& std::operator<< (
    basic_ostream<_CharT, _Traits> & __out, const _CharT* __s ) [inline]
```

String inserters.

Parameters

<code>__out</code>	An output stream.
<code>__s</code>	A character string.

Returns

out

Precondition

`__s` must be a non-NULL pointer

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts `traits::length(__s)` characters starting at `__s`, widened if necessary, followed by any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

Definition at line 539 of file `ostream`.

```
4.11.4.161 template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits> & std::operator<< (
    basic_ostream<_CharT, _Traits> & __out, const char * __s )
```

String inserters.

Parameters

<code>__out</code>	An output stream.
<code>__s</code>	A character string.

Returns

`out`

Precondition

`__s` must be a non-NULL pointer

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts `traits::length(__s)` characters starting at `__s`, widened if necessary, followed by any required padding (as determined by [22.2.2.2.2]). `__out.width(0)` is then called.

Definition at line 321 of file `ostream.tcc`.

References `std::ios_base::badbit`, `endl()`, `ends()`, `std::basic_ostream<_CharT, _Traits>::flush()`, `std::basic_ostream<_CharT, _Traits>::operator<<()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

```
4.11.4.162 template<class _Traits> basic_ostream<char, _Traits> & std::operator<< ( basic_ostream<char, _Traits> &
    __out, const char * __s ) [inline]
```

String inserters.

Parameters

<code>__out</code>	An output stream.
<code>__s</code>	A character string.

Returns

out

Precondition

___s must be a non-NULL pointer

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts `traits::length(___s)` characters starting at `___s`, widened if necessary, followed by any required padding (as determined by [22.2.2.2.2]). `___out.width(0)` is then called.

Definition at line 556 of file ostream.

4.11.4.163 `template<class _Traits> basic_ostream<char, _Traits>& std::operator<< (basic_ostream< char, _Traits> & ___out, const signed char * ___s) [inline]`

String inserters.

Parameters

<code>___out</code>	An output stream.
<code>___s</code>	A character string.

Returns

out

Precondition

___s must be a non-NULL pointer

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts `traits::length(___s)` characters starting at `___s`, widened if necessary, followed by any required padding (as determined by [22.2.2.2.2]). `___out.width(0)` is then called.

Definition at line 569 of file ostream.

4.11.4.164 `template<class _Traits> basic_ostream<char, _Traits>& std::operator<< (basic_ostream< char, _Traits> & ___out, const unsigned char * ___s) [inline]`

String inserters.

Parameters

<code>___out</code>	An output stream.
<code>___s</code>	A character string.

Returns

out

Precondition

___s must be a non-NULL pointer

Behaves like one of the formatted arithmetic inserters described in `std::basic_ostream`. After constructing a sentry object with good status, this function inserts `traits::length(___s)` characters starting at `___s`, widened if necessary, followed by any required padding (as determined by [22.2.2.2.2]). `___out.width(0)` is then called.

Definition at line 574 of file ostream.

```
4.11.4.165 template<typename _CharT, typename _Traits, typename _Tp> basic_ostream<_CharT, _Traits>&
std::operator<<( basic_ostream<_CharT, _Traits> && __os, const _Tp & __x ) [inline]
```

Generic inserter for rvalue stream.

Parameters

___os	An input stream.
___x	A reference to the object being inserted.

Returns

os

This is just a forwarding function to allow insertion to rvalue streams since they won't bind to the inserter functions that take an lvalue reference.

Definition at line 628 of file ostream.

```
4.11.4.166 template<class _CharT, class _Traits, size_t _Nb> std::basic_ostream<_CharT, _Traits>& std::operator<<(
std::basic_ostream<_CharT, _Traits> & __os, const bitset<_Nb> & __x )
```

Global I/O operators for bitsets.

Direct I/O between streams and bitsets is supported. Output is straightforward. Input will skip whitespace, only accept 0 and 1 characters, and will only extract as many digits as the bitset will hold.

Definition at line 1530 of file bitset.

```
4.11.4.167 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> basic_ostream<_CharT, _Traits>& std::operator<<( basic_ostream<_CharT, _Traits> & __os, const
__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base> & __str ) [inline]
```

Write string to a stream.

Parameters

<code>__os</code>	Output stream.
<code>__str</code>	String to write out.

Returns

Reference to the output stream.

Output characters of `__str` into `os` following the same rules as for writing a C string.

Definition at line 2630 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::data()`, `getline()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

```
4.11.4.168 template<typename _CharT, typename _Traits, typename _Alloc> basic_ostream<_CharT, _Traits>&
std::operator<<( basic_ostream<_CharT, _Traits> & __os, const basic_string<_CharT, _Traits, _Alloc> &
__str ) [inline]
```

Write string to a stream.

Parameters

<code>__os</code>	Output stream.
<code>__str</code>	String to write out.

Returns

Reference to the output stream.

Output characters of `__str` into `os` following the same rules as for writing a C string.

Definition at line 5325 of file `basic_string.h`.

References `std::basic_string<_CharT, _Traits, _Alloc>::data()`, `getline()`, and `std::basic_string<_CharT, _Traits, _Alloc>::size()`.

```
4.11.4.169 template<typename _Tp, typename _Seq> bool std::operator<=( const stack<_Tp, _Seq> & __x, const stack<
_Tp, _Seq> & __y ) [inline]
```

Based on `operator<`.

Definition at line 311 of file `stl_stack.h`.

4.11.4.170 `template<typename _Tp, typename _Seq> bool std::operator<= (const queue<_Tp, _Seq> &__x, const queue<_Tp, _Seq> &__y) [inline]`

Based on operator<.

Definition at line 334 of file `stl_queue.h`.

4.11.4.171 `template<typename _Key, typename _Compare, typename _Alloc> bool std::operator<= (const multiset<_Key, _Compare, _Alloc> &__x, const multiset<_Key, _Compare, _Alloc> &__y) [inline]`

Returns `!(y < x)`

Definition at line 861 of file `stl_multiset.h`.

4.11.4.172 `template<typename _Key, typename _Compare, typename _Alloc> bool std::operator<= (const set<_Key, _Compare, _Alloc> &__x, const set<_Key, _Compare, _Alloc> &__y) [inline]`

Returns `!(y < x)`

Definition at line 878 of file `stl_set.h`.

4.11.4.173 `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc> bool std::operator<= (const multimap<_Key, _Tp, _Compare, _Alloc> &__x, const multimap<_Key, _Tp, _Compare, _Alloc> &__y) [inline]`

Based on operator<.

Definition at line 1010 of file `stl_multimap.h`.

4.11.4.174 `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc> bool std::operator<= (const map<_Key, _Tp, _Compare, _Alloc> &__x, const map<_Key, _Tp, _Compare, _Alloc> &__y) [inline]`

Based on operator<.

Definition at line 1345 of file `stl_map.h`.

4.11.4.175 `template<typename _Tp, typename _Alloc> bool std::operator<= (const forward_list<_Tp, _Alloc> &__lx, const forward_list<_Tp, _Alloc> &__ly) [inline]`

Based on operator<.

Definition at line 1405 of file `forward_list.h`.

4.11.4.176 `template<typename _Tp, typename _Alloc> bool std::operator<= (const vector<_Tp, _Alloc> &__x, const vector<_Tp, _Alloc> &__y) [inline]`

Based on operator<.

Definition at line 1545 of file `stl_vector.h`.

4.11.4.177 `template<typename _Tp, typename _Alloc> bool std::operator<= (const list< _Tp, _Alloc> & __x, const list< _Tp, _Alloc> & __y) [inline]`

Based on operator<.

Definition at line 1906 of file stl_list.h.

4.11.4.178 `template<typename _Tp, typename _Alloc> bool std::operator<= (const deque< _Tp, _Alloc> & __x, const deque< _Tp, _Alloc> & __y) [inline]`

Based on operator<.

Definition at line 2260 of file stl_deque.h.

4.11.4.179 `template<typename _CharT, typename _Traits, typename _Alloc> bool std::operator<= (const basic_string< _CharT, _Traits, _Alloc> & __lhs, const basic_string< _CharT, _Traits, _Alloc> & __rhs) [inline], [noexcept]`

Test if string doesn't follow string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs` doesn't follow `__rhs`. False otherwise.

Definition at line 5211 of file basic_string.h.

References `std::basic_string< _CharT, _Traits, _Alloc>::compare()`.

4.11.4.180 `template<typename _CharT, typename _Traits, typename _Alloc> bool std::operator<= (const basic_string< _CharT, _Traits, _Alloc> & __lhs, const _CharT* __rhs) [inline]`

Test if string doesn't follow C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs` doesn't follow `__rhs`. False otherwise.

Definition at line 5224 of file basic_string.h.

References `std::basic_string<_CharT, _Traits, _Alloc>::compare()`.

4.11.4.181 `template<typename _CharT, typename _Traits, typename _Alloc> bool std::operator<= (const _CharT* __lhs, const basic_string<_CharT, _Traits, _Alloc> & __rhs) [inline]`

Test if C string doesn't follow string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__lhs` doesn't follow `__rhs`. False otherwise.

Definition at line 5236 of file basic_string.h.

4.11.4.182 `template<typename _StateT> bool std::operator== (const fpos<_StateT> & __lhs, const fpos<_StateT> & __rhs) [inline]`

Test if equivalent to another position.

Definition at line 216 of file postypes.h.

4.11.4.183 `template<typename _Tp> bool std::operator== (const _Fwd_list_iterator<_Tp> & __x, const _Fwd_list_const_iterator<_Tp> & __y) [inline], [noexcept]`

Forward list iterator equality comparison.

Definition at line 257 of file forward_list.h.

4.11.4.184 `template<typename _Tp, typename _Seq> bool std::operator== (const stack<_Tp, _Seq> & __x, const stack<_Tp, _Seq> & __y) [inline]`

Stack equality comparison.

Parameters

<code>__x</code>	A stack.
<code>__y</code>	A stack of the same type as <code>__x</code> .

Returns

True iff the size and elements of the stacks are equal.

This is an equivalence relation. Complexity and semantics depend on the underlying sequence type, but the expected rules are: this relation is linear in the size of the sequences, and stacks are considered equivalent if their sequences compare equal.

Definition at line 275 of file `stl_stack.h`.

```
4.11.4.185  template<typename _Tp, typename _Seq > bool std::operator== ( const queue< _Tp, _Seq > & __x, const queue<
    _Tp, _Seq > & __y ) [inline]
```

Queue equality comparison.

Parameters

$_x$	A queue.
$_y$	A queue of the same type as $_x$.

Returns

True iff the size and elements of the queues are equal.

This is an equivalence relation. Complexity and semantics depend on the underlying sequence type, but the expected rules are: this relation is linear in the size of the sequences, and queues are considered equivalent if their sequences compare equal.

Definition at line 298 of file `stl_queue.h`.

References `std::queue< _Tp, _Sequence >::c`.

```
4.11.4.186  template<typename _Key, typename _Compare, typename _Alloc > bool std::operator== ( const multiset< _Key,
    _Compare, _Alloc > & __x, const multiset< _Key, _Compare, _Alloc > & __y ) [inline]
```

Multiset equality comparison.

Parameters

$_x$	A multiset.
$_y$	A multiset of the same type as $_x$.

Returns

True iff the size and elements of the multisets are equal.

This is an equivalence relation. It is linear in the size of the multisets. Multisets are considered equivalent if their sizes are equal, and if corresponding elements compare equal.

Definition at line 823 of file `stl_multiset.h`.

```
4.11.4.187  template<typename _Key , typename _Compare , typename _Alloc > bool std::operator== ( const set< _Key,
    _Compare, _Alloc > & __x, const set< _Key, _Compare, _Alloc > & __y )  [inline]
```

Set equality comparison.

Parameters

$_x$	A set.
$_y$	A set of the same type as $_x$.

Returns

True iff the size and elements of the sets are equal.

This is an equivalence relation. It is linear in the size of the sets. Sets are considered equivalent if their sizes are equal, and if corresponding elements compare equal.

Definition at line 840 of file `stl_set.h`.

```
4.11.4.188  template<typename _Key , typename _Tp , typename _Compare , typename _Alloc > bool std::operator== ( const
    multimap< _Key, _Tp, _Compare, _Alloc > & __x, const multimap< _Key, _Tp, _Compare, _Alloc > & __y )
    [inline]
```

Multimap equality comparison.

Parameters

$_x$	A multimap.
$_y$	A multimap of the same type as $_x$.

Returns

True iff the size and elements of the maps are equal.

This is an equivalence relation. It is linear in the size of the multimaps. Multimaps are considered equivalent if their sizes are equal, and if corresponding elements compare equal.

Definition at line 972 of file stl_multimap.h.

Referenced by `std::multimap<_Key, _Tp, _Compare, _Alloc>::equal_range()`.

4.11.4.189 `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc> bool std::operator==(const map<_Key, _Tp, _Compare, _Alloc> &__x, const map<_Key, _Tp, _Compare, _Alloc> &__y) [inline]`

Map equality comparison.

Parameters

\leftrightarrow __x	A map.
\leftrightarrow __y	A map of the same type as x.

Returns

True iff the size and elements of the maps are equal.

This is an equivalence relation. It is linear in the size of the maps. Maps are considered equivalent if their sizes are equal, and if corresponding elements compare equal.

Definition at line 1307 of file stl_map.h.

Referenced by `std::map<_Key, _Tp, _Compare, _Alloc>::equal_range()`.

4.11.4.190 `template<typename _Tp, typename _Alloc> bool std::operator==(const forward_list<_Tp, _Alloc> &__lx, const forward_list<_Tp, _Alloc> &__ly)`

Forward list equality comparison.

Parameters

\leftrightarrow __lx	A forward_list
\leftrightarrow __ly	A forward_list of the same type as __lx.

Returns

True iff the elements of the forward lists are equal.

This is an equivalence relation. It is linear in the number of elements of the forward lists. Deques are considered equivalent if corresponding elements compare equal.

Definition at line 375 of file forward_list.tcc.

References `std::forward_list<_Tp, _Alloc>::cbegin()`, `std::forward_list<_Tp, _Alloc>::cend()`, and `std::forward_list<_Tp, _Alloc>::sort()`.

4.11.4.191 `template<typename _Tp, typename _Alloc> bool std::operator==(const vector<_Tp, _Alloc> & __x, const vector<_Tp, _Alloc> & __y) [inline]`

Vector equality comparison.

Parameters

<code>__x</code>	A vector.
<code>__y</code>	A vector of the same type as <code>__x</code> .

Returns

True iff the size and elements of the vectors are equal.

This is an equivalence relation. It is linear in the size of the vectors. Vectors are considered equivalent if their sizes are equal, and if corresponding elements compare equal.

Definition at line 1509 of file `std_vector.h`.

References `std::vector<_Tp, _Alloc>::begin()`, `std::vector<_Tp, _Alloc>::end()`, `equal()`, and `std::vector<_Tp, _Alloc>::size()`.

4.11.4.192 `template<typename _Tp, typename _Alloc> _GLIBCXX_END_NAMESPACE_CXX11 bool std::operator==(const list<_Tp, _Alloc> & __x, const list<_Tp, _Alloc> & __y) [inline]`

List equality comparison.

Parameters

<code>__x</code>	A list.
<code>__y</code>	A list of the same type as <code>__x</code> .

Returns

True iff the size and elements of the lists are equal.

This is an equivalence relation. It is linear in the size of the lists. Lists are considered equivalent if their sizes are equal, and if corresponding elements compare equal.

Definition at line 1853 of file `std_list.h`.

References `std::list<_Tp, _Alloc>::begin()`, `std::list<_Tp, _Alloc>::end()`, and `std::list<_Tp, _Alloc>::size()`.

4.11.4.193 `template<typename _Res, typename... _Args> bool std::operator==(const function< _Res(_Args...)> &__f, nullptr_t) [inline], [noexcept]`

Compares a polymorphic function object wrapper against 0 (the NULL pointer).

Returns

`true` if the wrapper has no target, `false` otherwise

This function will not throw an exception.

Definition at line 2193 of file functional.

4.11.4.194 `template<typename _Res, typename... _Args> bool std::operator==(nullptr_t, const function< _Res(_Args...)> &__f) [inline], [noexcept]`

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.

Definition at line 2199 of file functional.

4.11.4.195 `template<typename _Tp, typename _Alloc > bool std::operator==(const deque< _Tp, _Alloc > &__x, const deque< _Tp, _Alloc > &__y) [inline]`

Deque equality comparison.

Parameters

<code>__x</code>	A deque.
<code>__y</code>	A deque of the same type as <code>__x</code> .

Returns

True iff the size and elements of the deques are equal.

This is an equivalence relation. It is linear in the size of the deques. Deques are considered equivalent if their sizes are equal, and if corresponding elements compare equal.

Definition at line 2220 of file stl_deque.h.

References `std::deque< _Tp, _Alloc >::begin()`, `std::deque< _Tp, _Alloc >::end()`, `equal()`, and `std::deque< _Tp, _Alloc >::size()`.

4.11.4.196 `template<typename _CharT, typename _Traits, typename _Alloc > bool std::operator==(const basic_string< _CharT, _Traits, _Alloc > &__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs) [inline], [noexcept]`

Test equivalence of two strings.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs.compare(__rhs) == 0`. False otherwise.

Definition at line 5050 of file `basic_string.h`.

References `std::basic_string<_CharT, _Traits, _Alloc>::data()`, and `std::basic_string<_CharT, _Traits, _Alloc>::size()`.

4.11.4.197 `template<typename _CharT, typename _Traits, typename _Alloc> bool std::operator==(const _CharT* __lhs, const basic_string<_CharT, _Traits, _Alloc> & __rhs) [inline]`

Test equivalence of C string and string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__rhs.compare(__lhs) == 0`. False otherwise.

Definition at line 5072 of file `basic_string.h`.

4.11.4.198 `template<typename _CharT, typename _Traits, typename _Alloc> bool std::operator==(const basic_string<_CharT, _Traits, _Alloc> & __lhs, const _CharT* __rhs) [inline]`

Test equivalence of string and C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs.compare(__rhs) == 0`. False otherwise.

Definition at line 5084 of file `basic_string.h`.

4.11.4.199 `template<typename _Tp, typename _Seq> bool std::operator> (const stack< _Tp, _Seq> & __x, const stack< _Tp, _Seq> & __y) [inline]`

Based on operator<.

Definition at line 305 of file `stl_stack.h`.

4.11.4.200 `template<typename _Tp, typename _Seq> bool std::operator> (const queue< _Tp, _Seq> & __x, const queue< _Tp, _Seq> & __y) [inline]`

Based on operator<.

Definition at line 328 of file `stl_queue.h`.

4.11.4.201 `template<typename _Key, typename _Compare, typename _Alloc> bool std::operator> (const multiset< _Key, _Compare, _Alloc> & __x, const multiset< _Key, _Compare, _Alloc> & __y) [inline]`

Returns $y < x$.

Definition at line 854 of file `stl_multiset.h`.

4.11.4.202 `template<typename _Key, typename _Compare, typename _Alloc> bool std::operator> (const set< _Key, _Compare, _Alloc> & __x, const set< _Key, _Compare, _Alloc> & __y) [inline]`

Returns $y < x$.

Definition at line 871 of file `stl_set.h`.

4.11.4.203 `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc> bool std::operator> (const multimap< _Key, _Tp, _Compare, _Alloc> & __x, const multimap< _Key, _Tp, _Compare, _Alloc> & __y) [inline]`

Based on operator<.

Definition at line 1003 of file `stl_multimap.h`.

4.11.4.204 `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc> bool std::operator> (const map< _Key, _Tp, _Compare, _Alloc> & __x, const map< _Key, _Tp, _Compare, _Alloc> & __y) [inline]`

Based on operator<.

Definition at line 1338 of file `stl_map.h`.

4.11.4.205 `template<typename _Tp, typename _Alloc> bool std::operator> (const forward_list< _Tp, _Alloc> & __lx, const forward_list< _Tp, _Alloc> & __ly) [inline]`

Based on operator<.

Definition at line 1391 of file `forward_list.h`.

4.11.4.206 `template<typename _Tp, typename _Alloc > bool std::operator> (const vector<_Tp, _Alloc > &__x, const vector<_Tp, _Alloc > &__y) [inline]`

Based on operator<.

Definition at line 1539 of file `stl_vector.h`.

4.11.4.207 `template<typename _Tp, typename _Alloc > bool std::operator> (const list<_Tp, _Alloc > &__x, const list<_Tp, _Alloc > &__y) [inline]`

Based on operator<.

Definition at line 1900 of file `stl_list.h`.

4.11.4.208 `template<typename _Tp, typename _Alloc > bool std::operator> (const deque<_Tp, _Alloc > &__x, const deque<_Tp, _Alloc > &__y) [inline]`

Based on operator<.

Definition at line 2253 of file `stl_deque.h`.

4.11.4.209 `template<typename _CharT, typename _Traits, typename _Alloc > bool std::operator> (const basic_string<_CharT, _Traits, _Alloc > &__lhs, const basic_string<_CharT, _Traits, _Alloc > &__rhs) [inline], [noexcept]`

Test if string follows string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs` follows `__rhs`. False otherwise.

Definition at line 5173 of file `basic_string.h`.

4.11.4.210 `template<typename _CharT, typename _Traits, typename _Alloc > bool std::operator> (const basic_string<_CharT, _Traits, _Alloc > &__lhs, const _CharT* __rhs) [inline]`

Test if string follows C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs` follows `__rhs`. False otherwise.

Definition at line 5186 of file `basic_string.h`.

```
4.11.4.211 template<typename _CharT, typename _Traits, typename _Alloc > bool std::operator< ( const _CharT * __lhs, const
basic_string< _CharT, _Traits, _Alloc > & __rhs ) [inline]
```

Test if C string follows string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__lhs` follows `__rhs`. False otherwise.

Definition at line 5198 of file `basic_string.h`.

```
4.11.4.212 template<typename _Tp, typename _Seq > bool std::operator>= ( const stack< _Tp, _Seq > & __x, const stack<
_Tp, _Seq > & __y ) [inline]
```

Based on `operator<`.

Definition at line 317 of file `stl_stack.h`.

```
4.11.4.213 template<typename _Tp, typename _Seq > bool std::operator>= ( const queue< _Tp, _Seq > & __x, const
queue< _Tp, _Seq > & __y ) [inline]
```

Based on `operator<`.

Definition at line 340 of file `stl_queue.h`.

```
4.11.4.214 template<typename _Key, typename _Compare, typename _Alloc > bool std::operator>= ( const multiset< _Key,
_Compare, _Alloc > & __x, const multiset< _Key, _Compare, _Alloc > & __y ) [inline]
```

Returns `!(x < y)`

Definition at line 868 of file `stl_multiset.h`.

```
4.11.4.215 template<typename _Key, typename _Compare, typename _Alloc > bool std::operator>= ( const set< _Key,
_Compare, _Alloc > & __x, const set< _Key, _Compare, _Alloc > & __y ) [inline]
```

Returns `!(x < y)`

Definition at line 885 of file `stl_set.h`.

4.11.4.216 `template<typename _Key , typename _Tp , typename _Compare , typename _Alloc > bool std::operator>= (const multimap< _Key, _Tp, _Compare, _Alloc > &__x, const multimap< _Key, _Tp, _Compare, _Alloc > &__y) [inline]`

Based on operator<.

Definition at line 1017 of file `stl_multimap.h`.

4.11.4.217 `template<typename _Key , typename _Tp , typename _Compare , typename _Alloc > bool std::operator>= (const map< _Key, _Tp, _Compare, _Alloc > &__x, const map< _Key, _Tp, _Compare, _Alloc > &__y) [inline]`

Based on operator<.

Definition at line 1352 of file `stl_map.h`.

4.11.4.218 `template<typename _Tp , typename _Alloc > bool std::operator>= (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly) [inline]`

Based on operator<.

Definition at line 1398 of file `forward_list.h`.

4.11.4.219 `template<typename _Tp , typename _Alloc > bool std::operator>= (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y) [inline]`

Based on operator<.

Definition at line 1551 of file `stl_vector.h`.

4.11.4.220 `template<typename _Tp , typename _Alloc > bool std::operator>= (const list< _Tp, _Alloc > &__x, const list< _Tp, _Alloc > &__y) [inline]`

Based on operator<.

Definition at line 1912 of file `stl_list.h`.

4.11.4.221 `template<typename _Tp , typename _Alloc > bool std::operator>= (const deque< _Tp, _Alloc > &__x, const deque< _Tp, _Alloc > &__y) [inline]`

Based on operator<.

Definition at line 2267 of file `stl_deque.h`.

4.11.4.222 `template<typename _CharT , typename _Traits , typename _Alloc > bool std::operator>= (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs) [inline], [noexcept]`

Test if string doesn't precede string.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Returns

True if `__lhs` doesn't precede `__rhs`. False otherwise.

Definition at line 5249 of file `basic_string.h`.

```
4.11.4.223 template<typename _CharT, typename _Traits, typename _Alloc > bool std::operator>= ( const basic_string<
    _CharT, _Traits, _Alloc > & __lhs, const _CharT * __rhs ) [inline]
```

Test if string doesn't precede C string.

Parameters

<code>__lhs</code>	String.
<code>__rhs</code>	C string.

Returns

True if `__lhs` doesn't precede `__rhs`. False otherwise.

Definition at line 5262 of file `basic_string.h`.

```
4.11.4.224 template<typename _CharT, typename _Traits, typename _Alloc > bool std::operator>= ( const _CharT * __lhs, const
    basic_string< _CharT, _Traits, _Alloc > & __rhs ) [inline]
```

Test if C string doesn't precede string.

Parameters

<code>__lhs</code>	C string.
<code>__rhs</code>	String.

Returns

True if `__lhs` doesn't precede `__rhs`. False otherwise.

Definition at line 5274 of file `basic_string.h`.

```
4.11.4.225 template<typename _CharT, typename _Traits > basic_istream< _CharT, _Traits > & std::operator>> (
    basic_istream< _CharT, _Traits > & __in, _CharT & __c )
```

Character extractors.

Parameters

\leftrightarrow __in	An input stream.
\leftrightarrow __c	A character reference.

Returns

in

Behaves like one of the formatted arithmetic extractors described in `std::basic_istream`. After constructing a sentry object with good status, this function extracts a character (if one is available) and stores it in `__c`. Otherwise, sets failbit in the input stream.

Definition at line 923 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ios< \leftrightarrow _CharT, _Traits >::rdbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

4.11.4.226 `template<class _Traits> basic_istream<char, _Traits>& std::operator>> (basic_istream< char, _Traits> & __in, unsigned char & __c) [inline]`

Character extractors.

Parameters

\leftrightarrow __in	An input stream.
\leftrightarrow __c	A character reference.

Returns

in

Behaves like one of the formatted arithmetic extractors described in `std::basic_istream`. After constructing a sentry object with good status, this function extracts a character (if one is available) and stores it in `__c`. Otherwise, sets failbit in the input stream.

Definition at line 756 of file `istream`.

4.11.4.227 `template<class _Traits> basic_istream<char, _Traits>& std::operator>> (basic_istream< char, _Traits> & __in, signed char & __c) [inline]`

Character extractors.

Parameters

\leftrightarrow _in	An input stream.
\leftrightarrow _c	A character reference.

Returns

in

Behaves like one of the formatted arithmetic extractors described in `std::basic_istream`. After constructing a sentry object with good status, this function extracts a character (if one is available) and stores it in `__c`. Otherwise, sets failbit in the input stream.

Definition at line 761 of file `istream`.

4.11.4.228 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::operator>> (basic_istream<_CharT, _Traits> & __in, _CharT * __s)`

Character string extractors.

Parameters

\leftrightarrow _in	An input stream.
\leftrightarrow _s	A pointer to a character array.

Returns

__in

Behaves like one of the formatted arithmetic extractors described in `std::basic_istream`. After constructing a sentry object with good status, this function extracts up to `n` characters and stores them into the array starting at `__s`. `n` is defined as:

- if `width()` is greater than zero, `n` is `width()` otherwise
- `n` is *the number of elements of the largest array of **
- *char_type that can store a terminating eos.*
- `[27.6.1.2.3]/6`

Characters are extracted and stored until one of the following happens:

- `n-1` characters are stored

- EOF is reached
- the next character is whitespace according to the current locale
- the next character is a null byte (i.e., `charT()`)

`width(0)` is then called for the input stream.

If no characters are extracted, sets failbit.

Definition at line 955 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::getloc()`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::setstate()`, and `std::ios_base::width()`.

4.11.4.229 `template<> basic_istream<char>& std::operator>> (basic_istream<char> & __in, char * __s)`

Character string extractors.

Parameters

<code>__in</code>	An input stream.
<code>__s</code>	A pointer to a character array.

Returns

`__in`

Behaves like one of the formatted arithmetic extractors described in `std::basic_istream`. After constructing a sentry object with good status, this function extracts up to `n` characters and stores them into the array starting at `__s`. `n` is defined as:

- if `width()` is greater than zero, `n` is `width()` otherwise
- `n` is *the number of elements of the largest array of **
- *char_type that can store a terminating eos.*
- [27.6.1.2.3]/6

Characters are extracted and stored until one of the following happens:

- `n-1` characters are stored
- EOF is reached
- the next character is whitespace according to the current locale
- the next character is a null byte (i.e., `charT()`)

`width(0)` is then called for the input stream.

If no characters are extracted, sets failbit.

4.11.4.230 `template<class _Traits> basic_istream<char, _Traits>& std::operator>> (basic_istream< char, _Traits> & __in, unsigned char * __s) [inline]`

Character string extractors.

Parameters

<code>__in</code>	An input stream.
<code>__s</code>	A pointer to a character array.

Returns

`__in`

Behaves like one of the formatted arithmetic extractors described in `std::basic_istream`. After constructing a sentry object with good status, this function extracts up to `n` characters and stores them into the array starting at `__s`. `n` is defined as:

- if `width()` is greater than zero, `n` is `width()` otherwise
- `n` is *the number of elements of the largest array of **
- *char_type that can store a terminating eos.*
- [27.6.1.2.3]/6

Characters are extracted and stored until one of the following happens:

- `n-1` characters are stored
- EOF is reached
- the next character is whitespace according to the current locale
- the next character is a null byte (i.e., `charT()`)

`width(0)` is then called for the input stream.

If no characters are extracted, sets failbit.

Definition at line 803 of file `istream`.

4.11.4.231 `template<class _Traits> basic_istream<char, _Traits>& std::operator>> (basic_istream< char, _Traits> & __in, signed char * __s) [inline]`

Character string extractors.

Parameters

\leftarrow <code>_in</code>	An input stream.
\leftarrow <code>_s</code>	A pointer to a character array.

Returns

`__in`

Behaves like one of the formatted arithmetic extractors described in `std::basic_istream`. After constructing a sentry object with good status, this function extracts up to `n` characters and stores them into the array starting at `__s`. `n` is defined as:

- if `width()` is greater than zero, `n` is `width()` otherwise
- `n` is *the number of elements of the largest array of **
- *char_type that can store a terminating eos.*
- [27.6.1.2.3]/6

Characters are extracted and stored until one of the following happens:

- `n-1` characters are stored
- EOF is reached
- the next character is whitespace according to the current locale
- the next character is a null byte (i.e., `charT()`)

`width(0)` is then called for the input stream.

If no characters are extracted, sets failbit.

Definition at line 808 of file `istream`.

```
4.11.4.232 template<typename _CharT, typename _Traits, typename _Tp> basic_istream<_CharT, _Traits>&
std::operator>>( basic_istream<_CharT, _Traits> && __is, _Tp & __x ) [inline]
```

Generic extractor for rvalue stream.

Parameters

\leftarrow <code>_is</code>	An input stream.
\leftarrow <code>_x</code>	A reference to the extraction target.

Returns

`is`

This is just a forwarding function to allow extraction from rvalue streams since they won't bind to the extractor functions that take an lvalue reference.

Definition at line 924 of file `istream`.

```
4.11.4.233 template<class _CharT, class _Traits, size_t _Nb> std::basic_istream<_CharT, _Traits>& std::operator>> (
    std::basic_istream<_CharT, _Traits> & __is, bitset<_Nb> & __x )
```

Global I/O operators for bitsets.

Direct I/O between streams and bitsets is supported. Output is straightforward. Input will skip whitespace, only accept `0` and `1` characters, and will only extract as many digits as the bitset will hold.

Definition at line 1462 of file `bitset`.

Referenced by `getline()`, `operator>>()`, and `swap()`.

```
4.11.4.234 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
    _Base> basic_istream<_CharT, _Traits> & std::operator>> ( basic_istream<_CharT, _Traits> & __is,
    __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base> & __str )
```

Read stream into a string.

Parameters

<code>__is</code>	Input stream.
<code>__str</code>	Buffer to store into.

Returns

Reference to the input stream.

Stores characters from `__is` into `__str` until whitespace is found, the end of the stream is encountered, or `str.max_size()` is reached. If `is.width()` is non-zero, that is the limit on the number of characters stored into `__str`. Any previous contents of `__str` are erased.

Definition at line 552 of file `vstring.tcc`.

References `std::ios_base::getloc()`, `operator>>()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::setstate()`, and `std::ios_base::width()`.

```
4.11.4.235 template<typename _CharT, typename _Traits, typename _Alloc> basic_istream<_CharT, _Traits> &
    std::operator>> ( basic_istream<_CharT, _Traits> & __is, basic_string<_CharT, _Traits, _Alloc> & __str )
```

Read stream into a string.

Parameters

<code>__is</code>	Input stream.
<code>__str</code>	Buffer to store into.

Returns

Reference to the input stream.

Stores characters from `__is` into `__str` until whitespace is found, the end of the stream is encountered, or `str.max_size()` is reached. If `is.width()` is non-zero, that is the limit on the number of characters stored into `__str`. Any previous contents of `__str` are erased.

Definition at line 1437 of file `basic_string.tcc`.

References `std::basic_string<_CharT, _Traits, _Alloc>::append()`, `std::basic_string<_CharT, _Traits, _Alloc>::erase()`, `std::ios_base::getloc()`, `std::basic_string<_CharT, _Traits, _Alloc>::max_size()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::setstate()`, and `std::ios_base::width()`.

4.11.4.236 `template<size_t_Nb> bitset<_Nb> std::operator^ (const bitset<_Nb> &__x, const bitset<_Nb> &__y)`
`[inline], [noexcept]`

Global bitwise operations on bitsets.

Parameters

<code>__x</code>	A bitset.
<code>__y</code>	A bitset of the same size as <code>__x</code> .

Returns

A new bitset.

These should be self-explanatory.

Definition at line 1443 of file `bitset`.

4.11.4.237 `template<size_t_Nb> bitset<_Nb> std::operator| (const bitset<_Nb> &__x, const bitset<_Nb> &__y)`
`[inline], [noexcept]`

Global bitwise operations on bitsets.

Parameters

<code>__x</code>	A bitset.
<code>__y</code>	A bitset of the same size as <code>__x</code> .

Returns

A new bitset.

These should be self-explanatory.

Definition at line 1434 of file bitset.

4.11.4.238 `template<typename _InputIterator, typename _OutputIterator> _OutputIterator std::partial_sum (_InputIterator __first, _InputIterator __last, _OutputIterator __result)`

Return list of partial sums.

Accumulates the values in the range [first,last) using the + operator. As each successive input value is added into the total, that partial sum is written to `__result`. Therefore, the first value in `__result` is the first value of the input, the second value in `__result` is the sum of the first and second input values, and so on.

Parameters

<code>__first</code>	Start of input range.
<code>__last</code>	End of input range.
<code>__result</code>	Output sum.

Returns

Iterator pointing just beyond the values written to `__result`.

Definition at line 237 of file `stl_numeric.h`.

Referenced by `__gnu_parallel::__parallel_random_shuffle_drs_pu()`, `__gnu_parallel::__sequential_random_shuffle()`, and `operator>>()`.

4.11.4.239 `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation> _OutputIterator std::partial_sum (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _BinaryOperation __binary_op)`

Return list of partial sums.

Accumulates the values in the range [first,last) using `__binary_op`. As each successive input value is added into the total, that partial sum is written to `__result`. Therefore, the first value in `__result` is the first value of the input, the second value in `__result` is the sum of the first and second input values, and so on.

Parameters

<code>__first</code>	Start of input range.
<code>__last</code>	End of input range.
<code>__result</code>	Output sum.
<code>__binary_op</code>	Function object.

Returns

Iterator pointing just beyond the values written to `__result`.

Definition at line 278 of file `stl_numeric.h`.

```
4.11.4.240  template<typename _MoneyT > _Put_money<_MoneyT> std::put_money ( const _MoneyT & __mon, bool __intl =
            false ) [inline]
```

Extended manipulator for inserting money.

Parameters

<code>__mon</code>	Either long double or a specialization of <code>basic_string</code> .
<code>__intl</code>	A bool indicating whether international format is to be used.

Sent to a stream object, this manipulator inserts `__mon`.

Definition at line 306 of file `iomanip`.

```
4.11.4.241  template<typename _CharT > _Put_time<_CharT> std::put_time ( const std::tm * __tmb, const _CharT * __fmt )
            [inline]
```

Extended manipulator for formatting time.

This manipulator uses `time_put::put` to format time. [ext.manip]

Parameters

<code>__tmb</code>	struct <code>tm</code> time data to format.
<code>__fmt</code>	format string.

Definition at line 358 of file `iomanip`.

```
4.11.4.242  template<typename _CharT > auto std::quoted ( const _CharT * __string, _CharT __delim = _CharT('\"'),
            _CharT __escape = _CharT('\\') ) [inline]
```

Manipulator for quoted strings.

Parameters

<code>__string</code>	String to quote.
<code>__delim</code>	Character to quote string with.
<code>__escape</code>	Escape character to escape itself or quote character.

Definition at line 461 of file `iomanip`.

4.11.4.243 `template<typename _Container> auto std::rbegin (_Container & __cont)-> decltype(__cont.rbegin())` `[inline]`

Return a reverse iterator pointing to the last element of the container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

Definition at line 138 of file `range_access.h`.

Referenced by `crbegin()`, and `__gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >::node↔_end()`.

4.11.4.244 `template<typename _Container> auto std::rbegin (const _Container & __cont)-> decltype(__cont.rbegin())`
`[inline]`

Return a reverse iterator pointing to the last element of the const container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

Definition at line 148 of file `range_access.h`.

4.11.4.245 `template<typename _Tp, size_t _Nm> reverse_iterator<_Tp*> std::rbegin (_Tp(&) __arr[_Nm])` `[inline]`

Return a reverse iterator pointing to the last element of the array.

Parameters

<code>__arr</code>	Array.
--------------------	--------

Definition at line 178 of file `range_access.h`.

4.11.4.246 `template<typename _Tp> reverse_iterator<const _Tp*> std::rbegin (initializer_list<_Tp> __il)`
`[inline]`

Return a reverse iterator pointing to the last element of the `initializer_list`.

Parameters

<code>__il</code>	<code>initializer_list</code> .
-------------------	---------------------------------

Definition at line 198 of file `range_access.h`.

4.11.4.247 `template<typename _Tp> reference_wrapper<_Tp> std::ref (_Tp & __t) [inline], [noexcept]`

Denotes a reference should be taken to a variable.

Definition at line 473 of file functional.

4.11.4.248 `template<typename _Tp> void std::ref (const _Tp &&) [delete]`

Denotes a reference should be taken to a variable.

4.11.4.249 `template<typename _Tp> reference_wrapper<_Tp> std::ref (reference_wrapper<_Tp> __t) [inline], [noexcept]`

Partial specialization.

Definition at line 491 of file functional.

4.11.4.250 `template<typename _Container> auto std::rend (_Container & __cont)-> decltype(__cont.rend()) [inline]`

Return a reverse iterator pointing one past the first element of the container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

Definition at line 158 of file range_access.h.

Referenced by `crend()`, and `__gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >::node_↵end()`.

4.11.4.251 `template<typename _Container> auto std::rend (const _Container & __cont)-> decltype(__cont.rend()) [inline]`

Return a reverse iterator pointing one past the first element of the const container.

Parameters

<code>__cont</code>	Container.
---------------------	------------

Definition at line 168 of file range_access.h.

4.11.4.252 `template<typename _Tp, size_t _Nm> reverse_iterator<_Tp*> std::rend (_Tp(&) __arr[_Nm]) [inline]`

Return a reverse iterator pointing one past the first element of the array.

Parameters

<code>__arr</code>	Array.
--------------------	--------

Definition at line 188 of file `range_access.h`.

4.11.4.253 `template<typename _Tp > reverse_iterator<const _Tp*> std::rend (initializer_list<_Tp > __il)`
`[inline]`

Return a reverse iterator pointing one past the first element of the `initializer_list`.

Parameters

<code>__il</code>	<code>initializer_list</code> .
-------------------	---------------------------------

Definition at line 208 of file `range_access.h`.

4.11.4.254 `template<typename _InputIterator , typename _OutputIterator , typename _Tp > _OutputIterator std::replace_copy (`
`_InputIterator __first, _InputIterator __last, _OutputIterator __result, const _Tp & __old_value, const _Tp & __new_value`
`) [inline]`

Copy a sequence, replacing each element of one value with another value.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.
<code>__old_value</code>	The value to be replaced.
<code>__new_value</code>	The replacement value.

Returns

The end of the output sequence, `result+(last-first)`.

Copies each element in the input range `[__first,__last)` to the output range `[__result,__result+(__last-__first))` replacing elements equal to `__old_value` with `__new_value`.

Definition at line 3133 of file `stl_algo.h`.

4.11.4.255 `_Resetiosflags std::resetiosflags (ios_base::fmtflags __mask) [inline]`

Manipulator for `setf`.

Parameters

<code>__mask</code>	A format flags mask.
---------------------	----------------------

Sent to a stream object, this manipulator resets the specified flags, via `stream.setf(0,__mask)`.

Definition at line 66 of file `iomanip`.

4.11.4.256 `template<typename _Tp> void std::return_temporary_buffer (_Tp* __p) [inline]`

The companion to `get_temporary_buffer()`.

Parameters

<code>__p</code>	A buffer previously allocated by <code>get_temporary_buffer</code> .
------------------	--

Returns

None.

Frees the memory pointed to by `__p`.

Definition at line 112 of file `stl_tempbuf.h`.

Referenced by `std::Temporary_buffer<_ForwardIterator, _Tp>::_Temporary_buffer()`, and `std::Temporary_buffer<_ForwardIterator, _Tp>::end()`.

4.11.4.257 `ios_base& std::right (ios_base & __base) [inline]`

Calls `base.setf(ios_base::right, ios_base::adjustfield)`.

Definition at line 1007 of file `ios_base.h`.

References `std::ios_base::adjustfield`, `std::ios_base::right`, and `std::ios_base::setf()`.

4.11.4.258 `ios_base& std::scientific (ios_base & __base) [inline]`

Calls `base.setf(ios_base::scientific, ios_base::floatfield)`.

Definition at line 1049 of file `ios_base.h`.

References `std::ios_base::floatfield`, `std::ios_base::scientific`, and `std::ios_base::setf()`.

Referenced by `std::normal_distribution<_RealType>::operator()()`, `std::gamma_distribution<_RealType>::operator()()`, `std::binomial_distribution<_IntType>::operator()()`, `std::negative_binomial_distribution<_IntType>::operator()()`, `std::poisson_distribution<_IntType>::operator()()`, `operator<<()`, and `operator>>()`.

4.11.4.259 `new_handler std::set_new_handler (new_handler) throw`

Takes a replacement handler as the argument, returns the previous handler.

4.11.4.260 `_Setbase std::setbase (int __base) [inline]`

Manipulator for `setf`.

Parameters

<code>__base</code>	A numeric base.
---------------------	-----------------

Sent to a stream object, this manipulator changes the `ios_base::basefield` flags to `oct`, `dec`, or `hex` when `base` is 8, 10, or 16, accordingly, and to 0 if `__base` is any other value.

Definition at line 127 of file `iomanip`.

4.11.4.261 `template<typename _CharT> _Setfill<_CharT> std::setfill (_CharT __c) [inline]`

Manipulator for `fill`.

Parameters

<code>__c</code>	The new fill character.
------------------	-------------------------

Sent to a stream object, this manipulator calls `fill (__c)` for that object.

Definition at line 165 of file `iomanip`.

4.11.4.262 `_Setiosflags std::setiosflags (ios_base::fmtflags __mask) [inline]`

Manipulator for `setf`.

Parameters

<code>__mask</code>	A format flags mask.
---------------------	----------------------

Sent to a stream object, this manipulator sets the format flags to `__mask`.

Definition at line 96 of file `iomanip`.

4.11.4.263 `_Setprecision std::setprecision (int __n) [inline]`

Manipulator for `precision`.

Parameters

<code>__n</code>	The new precision.
------------------	--------------------

Sent to a stream object, this manipulator calls `precision (__n)` for that object.

Definition at line 195 of file `iomanip`.

4.11.4.264 `_Setw` `std::setw (int __n) [inline]`

Manipulator for `width`.

Parameters

<code>__n</code>	The new width.
------------------	----------------

Sent to a stream object, this manipulator calls `width (__n)` for that object.

Definition at line 225 of file `iosmanip`.

4.11.4.265 `ios_base& std::showbase (ios_base & __base) [inline]`

Calls `base.setf(ios_base::showbase)`.

Definition at line 894 of file `ios_base.h`.

References `std::ios_base::setf()`, and `std::ios_base::showbase`.

4.11.4.266 `ios_base& std::showpoint (ios_base & __base) [inline]`

Calls `base.setf(ios_base::showpoint)`.

Definition at line 910 of file `ios_base.h`.

References `std::ios_base::setf()`, and `std::ios_base::showpoint`.

4.11.4.267 `ios_base& std::showpos (ios_base & __base) [inline]`

Calls `base.setf(ios_base::showpos)`.

Definition at line 926 of file `ios_base.h`.

References `std::ios_base::setf()`, and `std::ios_base::showpos`.

4.11.4.268 `ios_base& std::skipws (ios_base & __base) [inline]`

Calls `base.setf(ios_base::skipws)`.

Definition at line 942 of file `ios_base.h`.

References `std::ios_base::setf()`, and `std::ios_base::skipws`.

Referenced by `std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >::discard()`, `std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::operator()()`, `std::discard_block_engine< _RandomNumberEngine, __p, __r >::operator()()`, `std::shuffle_order_engine< _RandomNumberEngine, __k >::operator()()`, `std::normal_distribution< _RealType >::operator()()`, `std::gamma_distribution< _RealType >::operator()()`, `std::binomial_distribution< _IntType >::operator()()`, `std::negative_binomial_distribution< _IntType >::operator()()`, `std::poisson_distribution< _IntType >::operator()()`, `std::__detail::operator>>()`, and `operator>>()`.

4.11.4.269 `template<typename _Tp, typename _Tp1, _Lock_policy _Lp> __shared_ptr<_Tp, _Lp> std::static_pointer_cast (const __shared_ptr<_Tp1, _Lp> &__r) [inline], [noexcept]`

static_pointer_cast

Definition at line 1314 of file shared_ptr_base.h.

4.11.4.270 `template<class _CharT, class _Traits, class _Allocator> void std::swap (basic_stringbuf<_CharT, _Traits, _Allocator> &__x, basic_stringbuf<_CharT, _Traits, _Allocator> &__y) [inline]`

Swap specialization for stringbufs.

Definition at line 783 of file sstream.

4.11.4.271 `template<class _CharT, class _Traits, class _Allocator> void std::swap (basic_istream<_CharT, _Traits, _Allocator> &__x, basic_istream<_CharT, _Traits, _Allocator> &__y) [inline]`

Swap specialization for istreams.

Definition at line 790 of file sstream.

4.11.4.272 `template<class _CharT, class _Traits, class _Allocator> void std::swap (basic_ostringstream<_CharT, _Traits, _Allocator> &__x, basic_ostringstream<_CharT, _Traits, _Allocator> &__y) [inline]`

Swap specialization for ostringstreams.

Definition at line 797 of file sstream.

4.11.4.273 `template<class _CharT, class _Traits, class _Allocator> void std::swap (basic_stringstream<_CharT, _Traits, _Allocator> &__x, basic_stringstream<_CharT, _Traits, _Allocator> &__y) [inline]`

Swap specialization for stringstreams.

Definition at line 804 of file sstream.

4.11.4.274 `template<typename _Key, typename _Compare, typename _Alloc> void std::swap (multiset<_Key, _Compare, _Alloc> &__x, multiset<_Key, _Compare, _Alloc> &__y) [inline], [noexcept]`

See std::multiset::swap().

Definition at line 875 of file stl_multiset.h.

4.11.4.275 `template<typename _Key, typename _Compare, typename _Alloc> void std::swap (set<_Key, _Compare, _Alloc> &__x, set<_Key, _Compare, _Alloc> &__y) [inline], [noexcept]`

See std::set::swap().

Definition at line 892 of file stl_set.h.

4.11.4.276 `template<typename _Key , typename _Tp , typename _Compare , typename _Alloc > void std::swap (multimap< _Key, _Tp, _Compare, _Alloc > & __x, multimap< _Key, _Tp, _Compare, _Alloc > & __y) [inline], [noexcept]`

See `std::multimap::swap()`.

Definition at line 1024 of file `stl_multimap.h`.

4.11.4.277 `template<class _CharT , class _Traits > void std::swap (basic_filebuf< _CharT, _Traits > & __x, basic_filebuf< _CharT, _Traits > & __y) [inline]`

Swap specialization for filebufs.

Definition at line 1052 of file `fstream`.

4.11.4.278 `template<class _CharT , class _Traits > void std::swap (basic_ifstream< _CharT, _Traits > & __x, basic_ifstream< _CharT, _Traits > & __y) [inline]`

Swap specialization for ifstreams.

Definition at line 1059 of file `fstream`.

4.11.4.279 `template<class _CharT , class _Traits > void std::swap (basic_ofstream< _CharT, _Traits > & __x, basic_ofstream< _CharT, _Traits > & __y) [inline]`

Swap specialization for ofstreams.

Definition at line 1066 of file `fstream`.

4.11.4.280 `template<class _CharT , class _Traits > void std::swap (basic_fstream< _CharT, _Traits > & __x, basic_fstream< _CharT, _Traits > & __y) [inline]`

Swap specialization for fstreams.

Definition at line 1073 of file `fstream`.

4.11.4.281 `template<typename _Key , typename _Tp , typename _Compare , typename _Alloc > void std::swap (map< _Key, _Tp, _Compare, _Alloc > & __x, map< _Key, _Tp, _Compare, _Alloc > & __y) [inline], [noexcept]`

See `std::map::swap()`.

Definition at line 1359 of file `stl_map.h`.

4.11.4.282 `template<typename _Tp , typename _Alloc > void std::swap (forward_list< _Tp, _Alloc > & __lx, forward_list< _Tp, _Alloc > & __ly) [inline], [noexcept]`

See `std::forward_list::swap()`.

Definition at line 1412 of file `forward_list.h`.

4.11.4.283 `template<typename _Tp, typename _Alloc > void std::swap (vector< _Tp, _Alloc > &__x, vector< _Tp, _Alloc > &__y) [inline], [noexcept]`

See `std::vector::swap()`.

Definition at line 1557 of file `stl_vector.h`.

4.11.4.284 `template<typename _Tp, typename _Alloc > void std::swap (list< _Tp, _Alloc > &__x, list< _Tp, _Alloc > &__y) [inline], [noexcept]`

See `std::list::swap()`.

Definition at line 1918 of file `stl_list.h`.

4.11.4.285 `template<typename _Res, typename... _Args> void std::swap (function< _Res(_Args...)> &__x, function< _Res(_Args...)> &__y) [inline], [noexcept]`

Swap the targets of two polymorphic function object wrappers.

This function will not throw an exception.

Definition at line 2231 of file `functional`.

4.11.4.286 `template<typename _Tp, typename _Alloc > void std::swap (deque< _Tp, _Alloc > &__x, deque< _Tp, _Alloc > &__y) [inline], [noexcept]`

See `std::deque::swap()`.

Definition at line 2274 of file `stl_deque.h`.

4.11.4.287 `template<typename _CharT, typename _Traits, typename _Alloc > void std::swap (basic_string< _CharT, _Traits, _Alloc > &__lhs, basic_string< _CharT, _Traits, _Alloc > &__rhs) [inline], [noexcept]`

Swap contents of two strings.

Parameters

<code>__lhs</code>	First string.
<code>__rhs</code>	Second string.

Exchanges the contents of `__lhs` and `__rhs` in constant time.

Definition at line 5287 of file `basic_string.h`.

References operator `>>()`.

4.11.4.288 `template<typename _CharT > _CharT std::tolower (_CharT __c, const locale & __loc) [inline]`

Convenience interface to `ctype::tolower(__c)`.

Definition at line 2645 of file `locale_facets.h`.

References `std::__ctype_abstract_base< _CharT >::tolower()`.

4.11.4.289 `template<typename _CharT > _CharT std::toupper (_CharT __c, const locale & __loc) [inline]`

Convenience interface to `ctype::toupper(__c)`.

Definition at line 2639 of file `locale_facets.h`.

References `std::__ctype_abstract_base< _CharT >::toupper()`.

4.11.4.290 `template<typename _Lock1 , typename _Lock2 , typename... _Lock3> int std::try_lock (_Lock1 & __l1, _Lock2 & __l2, _Lock3 &... __l3)`

Generic `try_lock`.

Parameters

<code>__l1</code>	Meets Mutex requirements (<code>try_lock()</code> may throw).
<code>__l2</code>	Meets Mutex requirements (<code>try_lock()</code> may throw).
<code>__l3</code>	Meets Mutex requirements (<code>try_lock()</code> may throw).

Returns

Returns -1 if all `try_lock()` calls return true. Otherwise returns a 0-based index corresponding to the argument that returned false.

Postcondition

Either all arguments are locked, or none will be.

Sequentially calls `try_lock()` on each argument.

Definition at line 519 of file `mutex`.

4.11.4.291 `template<typename _InputIterator , typename _ForwardIterator > _ForwardIterator std::uninitialized_copy (_InputIterator __first, _InputIterator __last, _ForwardIterator __result) [inline]`

Copies the range `[first,last)` into `result`.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__result</code>	An output iterator.

Returns

`__result + (__first - __last)`

Like `copy()`, but does not require an initialized output range.

Definition at line 107 of file `stl_uninitialized.h`.

References `__addressof()`, `_Construct()`, and `_Destroy()`.

Referenced by `__gnu_parallel::parallel_sort_mwms_pu()`, and `uninitialized_fill_n()`.

4.11.4.292 `template<typename _InputIterator, typename _Size, typename _ForwardIterator> _ForwardIterator
std::uninitialized_copy_n(_InputIterator __first, _Size __n, _ForwardIterator __result) [inline]`

Copies the range `[first,first+n)` into `result`.

Parameters

<code>__first</code>	An input iterator.
<code>__n</code>	The number of elements to copy.
<code>__result</code>	An output iterator.

Returns

`__result + __n`

Like `copy_n()`, but does not require an initialized output range.

Definition at line 679 of file `stl_uninitialized.h`.

References `__iterator_category()`.

4.11.4.293 `template<typename _ForwardIterator, typename _Tp> void std::uninitialized_fill(_ForwardIterator __first,
_ForwardIterator __last, const _Tp & __x) [inline]`

Copies the value `x` into the range `[first,last)`.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__x</code>	The source value.

Returns

Nothing.

Like fill(), but does not require an initialized output range.

Definition at line 173 of file stl_uninitialized.h.

References __addressof(), _Construct(), _Destroy(), and fill_n().

Referenced by uninitialized_fill_n().

4.11.4.294 `template<typename _ForwardIterator, typename _Size, typename _Tp> _ForwardIterator std::uninitialized_fill_n (`
`_FowardIterator __first, _Size __n, const _Tp &__x) [inline]`

Copies the value x into the range [first,first+n).

Parameters

<code>__first</code>	An input iterator.
<code>__n</code>	The number of copies to make.
<code>__x</code>	The source value.

Returns

Nothing.

Like fill_n(), but does not require an initialized output range.

Definition at line 236 of file stl_uninitialized.h.

References __addressof(), _Construct(), _Destroy(), fill_n(), uninitialized_copy(), and uninitialized_fill().

4.11.4.295 `ios_base& std::unitbuf (ios_base &__base) [inline]`

Calls base.setf(ios_base::unitbuf).

Definition at line 974 of file ios_base.h.

References std::ios_base::setf(), and std::ios_base::unitbuf.

4.11.4.296 `ios_base& std::uppercase (ios_base &__base) [inline]`

Calls base.setf(ios_base::uppercase).

Definition at line 958 of file ios_base.h.

References std::ios_base::setf(), and std::ios_base::uppercase.

4.11.4.297 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::ws (basic_istream< _CharT, _Traits> & __is)`

Quick and easy way to eat whitespace.

This manipulator extracts whitespace characters, stopping when the next character is non-whitespace, or when the input sequence is empty. If the sequence is empty, `eofbit` is set in the stream, but not `failbit`.

The current locale is used to distinguish whitespace characters.

Example:

```
MyClass mc;
std::cin >> std::ws >> mc;
```

will skip leading whitespace before calling `operator>>` on `cin` and your object. Note that the same effect can be achieved by creating a `std::basic_istream::sentry` inside your definition of `operator>>`.

Definition at line 1016 of file `istream.tcc`.

References `std::ios_base::eofbit`, `std::ios_base::getloc()`, `std::basic_istream< _CharT, _Traits>::operator>>()`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

4.11.5 Variable Documentation

4.11.5.1 `ios_base::init std::__ioinit` `[static]`

Linked to standard error (buffered)

Definition at line 74 of file `iostream`.

4.11.5.2 `__thread void(* std::__once_call) ()`

Generic `try_lock`.

Parameters

<code>__l1</code>	Meets Mutex requirements (<code>try_lock()</code> may throw).
<code>__l2</code>	Meets Mutex requirements (<code>try_lock()</code> may throw).
<code>__l3</code>	Meets Mutex requirements (<code>try_lock()</code> may throw).

Returns

Returns -1 if all `try_lock()` calls return true. Otherwise returns a 0-based index corresponding to the argument that returned false.

Postcondition

Either all arguments are locked, or none will be.

Sequentially calls `try_lock()` on each argument.

4.11.5.3 `__thread void* std::__once_callable`

Generic `try_lock`.

Parameters

\leftrightarrow _l1	Meets Mutex requirements (<code>try_lock()</code> may throw).
\leftrightarrow _l2	Meets Mutex requirements (<code>try_lock()</code> may throw).
\leftrightarrow _l3	Meets Mutex requirements (<code>try_lock()</code> may throw).

Returns

Returns -1 if all `try_lock()` calls return true. Otherwise returns a 0-based index corresponding to the argument that returned false.

Postcondition

Either all arguments are locked, or none will be.

Sequentially calls `try_lock()` on each argument.

4.11.5.4 `ostream std::cerr`

Linked to standard output.

4.11.5.5 `istream std::cin`

Linked to standard input.

4.11.5.6 `ostream std::clog`

Linked to standard error (unbuffered)

4.11.5.7 `ostream` `std::cout`

Linked to standard input.

4.11.5.8 `wostream` `std::wcerr`

Linked to standard output.

4.11.5.9 `wistream` `std::wcin`

Linked to standard error (buffered)

4.11.5.10 `wostream` `std::wclog`

Linked to standard error (unbuffered)

4.11.5.11 `wostream` `std::wcout`

Linked to standard input.

4.12 `std::__debug` Namespace Reference

Classes

- class [bitset](#)
- class [deque](#)
- class [forward_list](#)
- class [list](#)
- class [map](#)
- class [multimap](#)
- class [multiset](#)
- class [set](#)
- class [unordered_map](#)
- class [unordered_multimap](#)
- class [unordered_multiset](#)
- class [unordered_set](#)
- class [vector](#)

Functions

- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>`
`constexpr _Tp & get (array< _Tp, _Nm > &__arr) noexcept`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>`
`constexpr _Tp && get (array< _Tp, _Nm > &&__arr) noexcept`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>`
`constexpr const _Tp & get (const array< _Tp, _Nm > &__arr) noexcept`
- `template<typename _Tp, std::size_t _Nm>`
`bool operator!= (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc >`
`bool operator!= (const unordered_set< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_set< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator!= (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator!= (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator!= (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc >`
`bool operator!= (const unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, const unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool operator!= (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator!= (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator!= (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator!= (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator!= (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc >`
`bool operator!= (const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc >`
`bool operator!= (const unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, const unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__y)`
- `template<size_t _Nb>`
`bitset< _Nb > operator& (const bitset< _Nb > &__x, const bitset< _Nb > &__y) noexcept`
- `template<typename _Tp, std::size_t _Nm>`
`bool operator< (const array< _Tp, _Nm > &__a, const array< _Tp, _Nm > &__b)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator< (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator< (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator > &__rhs)`

- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator< (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator< (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator< (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator< (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator< (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Tp, typename _Alloc >`
`bool operator< (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, size_t _Nb>`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const bitset< _Nb > &__x)`
- `template<typename _Tp, std::size_t _Nm>`
`bool operator<= (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator<= (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator<= (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator<= (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator<= (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator<= (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator<= (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator<= (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator<= (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Tp, std::size_t _Nm>`
`bool operator== (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc >`
`bool operator== (const unordered_set< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_set< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator== (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator== (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator== (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp, _Compare, _Allocator > &__rhs)`

- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc >`
`bool operator== (const unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, const unordered_map<`
`_Key, _Tp, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool operator== (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator== (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare,`
`_Allocator > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator== (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator== (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator== (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc >`
`bool operator== (const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_multiset<`
`_Value, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc >`
`bool operator== (const unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, const unordered_↵`
`multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Tp, std::size_t _Nm>`
`bool operator> (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator> (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, ↵`
`_Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator> (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator >`
`&__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator> (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp,`
`_Compare, _Allocator > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator> (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator> (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare,`
`_Allocator > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator> (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator> (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Tp, typename _Alloc >`
`bool operator> (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, std::size_t _Nm>`
`bool operator>= (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator>= (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare,`
`_Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator>= (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator >`
`&__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator>= (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp,`
`_Compare, _Allocator > &__rhs)`

- `template<typename _Tp, typename _Alloc >`
`bool operator>= (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator>= (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator>= (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator>= (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator>= (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _CharT, typename _Traits, size_t _Nb>`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, bitset< _Nb > &__x)`
- `template<size_t _Nb>`
`bitset< _Nb > operator^ (const bitset< _Nb > &__x, const bitset< _Nb > &__y) noexcept`
- `template<size_t _Nb>`
`bitset< _Nb > operator| (const bitset< _Nb > &__x, const bitset< _Nb > &__y) noexcept`
- `template<typename _Tp, std::size_t _Nm>`
`void swap (array< _Tp, _Nm > &__one, array< _Tp, _Nm > &__two) noexcept(noexcept(__one.swap(__two)))`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc >`
`void swap (unordered_set< _Value, _Hash, _Pred, _Alloc > &__x, unordered_set< _Value, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`void swap (multiset< _Key, _Compare, _Allocator > &__x, multiset< _Key, _Compare, _Allocator > &__y) noexcept(*conditional *)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`void swap (set< _Key, _Compare, _Allocator > &__x, set< _Key, _Compare, _Allocator > &__y) noexcept(*conditional *)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`void swap (multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, multimap< _Key, _Tp, _Compare, _Allocator > &__rhs) noexcept(*conditional *)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc >`
`void swap (unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Tp, typename _Alloc >`
`void swap (deque< _Tp, _Alloc > &__lhs, deque< _Tp, _Alloc > &__rhs) noexcept(*conditional *)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`void swap (map< _Key, _Tp, _Compare, _Allocator > &__lhs, map< _Key, _Tp, _Compare, _Allocator > &__rhs) noexcept(*conditional *)`
- `template<typename _Tp, typename _Alloc >`
`void swap (vector< _Tp, _Alloc > &__lhs, vector< _Tp, _Alloc > &__rhs) noexcept(*conditional *)`
- `template<typename _Tp, typename _Alloc >`
`void swap (list< _Tp, _Alloc > &__lhs, list< _Tp, _Alloc > &__rhs) noexcept(*conditional *)`
- `template<typename _Tp, typename _Alloc >`
`void swap (forward_list< _Tp, _Alloc > &__lx, forward_list< _Tp, _Alloc > &__ly) noexcept(noexcept(__lx.swap(__ly)))`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc >`
`void swap (unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc >`
`void swap (unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`

4.12.1 Detailed Description

GNU debug code, replaces standard behavior with debug behavior.

Macros and namespaces used by the implementation outside of debug wrappers to verify certain properties. The `__glibcxx_requires_xxx` macros are merely wrappers around the `__glibcxx_check_xxx` wrappers when we are compiling with debug mode, but disappear when we are in release mode so that there is no checking performed in, e.g., the standard library algorithms.

4.12.2 Function Documentation

4.12.2.1 `template<typename _Tp, typename _Alloc> bool std::__debug::operator<= (const forward_list<_Tp, _Alloc> & __lx, const forward_list<_Tp, _Alloc> & __ly) [inline]`

Based on operator<.

Definition at line 810 of file debug/forward_list.

4.12.2.2 `template<typename _Tp, typename _Alloc> bool std::__debug::operator> (const forward_list<_Tp, _Alloc> & __lx, const forward_list<_Tp, _Alloc> & __ly) [inline]`

Based on operator<.

Definition at line 796 of file debug/forward_list.

4.12.2.3 `template<typename _Tp, typename _Alloc> bool std::__debug::operator>= (const forward_list<_Tp, _Alloc> & __lx, const forward_list<_Tp, _Alloc> & __ly) [inline]`

Based on operator<.

Definition at line 803 of file debug/forward_list.

4.12.2.4 `template<typename _Tp, typename _Alloc> void std::__debug::swap (forward_list<_Tp, _Alloc> & __lx, forward_list<_Tp, _Alloc> & __ly) [inline], [noexcept]`

See `std::forward_list::swap()`.

Definition at line 817 of file debug/forward_list.

4.13 std::__detail Namespace Reference

Classes

- struct [_BracketMatcher](#)
- class [_Compiler](#)
- struct [_Default_ranged_hash](#)
- struct [_Equal_helper](#)
- struct [_Equal_helper< _Key, _Value, _ExtractKey, _Equal, _HashCodeType, false >](#)
- struct [_Equal_helper< _Key, _Value, _ExtractKey, _Equal, _HashCodeType, true >](#)
- struct [_Equality](#)
- struct [_Equality< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, false >](#)
- struct [_Equality< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, true >](#)
- struct [_Equality_base](#)
- class [_Executor](#)
- struct [_Hash_code_base](#)
- struct [_Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Default_ranged_hash, false >](#)
- struct [_Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Default_ranged_hash, true >](#)
- struct [_Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, false >](#)
- struct [_Hash_node](#)
- struct [_Hash_node< _Value, false >](#)
- struct [_Hash_node< _Value, true >](#)
- struct [_Hash_node_base](#)
- struct [_Hash_node_value_base](#)
- struct [_Hashtable_alloc](#)
- struct [_Hashtable_base](#)
- struct [_Hashtable_ebo_helper](#)
- struct [_Hashtable_ebo_helper< _Nm, _Tp, false >](#)
- struct [_Hashtable_ebo_helper< _Nm, _Tp, true >](#)
- struct [_Hashtable_traits](#)
- struct [_Insert](#)
- struct [_Insert< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, false, _↵ Unique_keys >](#)
- struct [_Insert< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, true, false >](#)
- struct [_Insert< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, true, true >](#)
- struct [_Insert_base](#)
- struct [_List_node_base](#)
- struct [_Local_const_iterator](#)
- struct [_Local_iterator](#)
- struct [_Local_iterator_base](#)
- struct [_Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, true >](#)
- struct [_Map_base](#)
- struct [_Map_base< _Key, _Pair, _Alloc, _Select1st, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, false >](#)
- struct [_Map_base< _Key, _Pair, _Alloc, _Select1st, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, true >](#)
- struct [_Mod_range_hashing](#)
- struct [_Node_const_iterator](#)
- struct [_Node_iterator](#)
- struct [_Node_iterator_base](#)
- struct [_Prime_rehash_policy](#)
- struct [_Quoted_string](#)

- struct [_Rehash_base](#)
- struct [_Rehash_base](#)< [_Key](#), [_Value](#), [_Alloc](#), [_ExtractKey](#), [_Equal](#), [_H1](#), [_H2](#), [_Hash](#), [_Prime_rehash_policy](#), [_Traits](#) >
- class [_Scanner](#)
- class [_StateSeq](#)

Typedefs

- template<typename [_Iter](#), typename [_TraitsT](#) >
using [__disable_if_contiguous_normal_iter](#) = typename enable_if< ![__is_contiguous_normal_iter](#)< [_Iter](#) >↵
::value, [std::shared_ptr](#)< const [_NFA](#)< [_TraitsT](#) >> >::type
- template<typename [_Iter](#), typename [_TraitsT](#) >
using [__enable_if_contiguous_normal_iter](#) = typename enable_if< [__is_contiguous_normal_iter](#)< [_Iter](#) >↵
::value, [std::shared_ptr](#)< const [_NFA](#)< [_TraitsT](#) >> >::type
- template<typename [_Key](#), typename [_Value](#), typename [_ExtractKey](#), typename [_H1](#), typename [_H2](#), typename [_Hash](#) >
using [__hash_code_for_local_iter](#) = [_Hash_code_storage](#)< [_Hash_code_base](#)< [_Key](#), [_Value](#), [_ExtractKey](#),
[_H1](#), [_H2](#), [_Hash](#), false >>
- template<typename [_CharT](#) >
using [_Matcher](#) = std::function< bool([_CharT](#))>
- typedef long [_StateIdT](#)

Enumerations

- enum [_Opcode](#) : int {
[_S_opcode_unknown](#), [_S_opcode_alternative](#), [_S_opcode_repeat](#), [_S_opcode_backref](#),
[_S_opcode_line_begin_assertion](#), [_S_opcode_line_end_assertion](#), [_S_opcode_word_boundary](#), [_S_](#)↵
[opcode_subexpr_lookahead](#),
[_S_opcode_subexpr_begin](#), [_S_opcode_subexpr_end](#), [_S_opcode_dummy](#), [_S_opcode_match](#),
[_S_opcode_accept](#) }
- enum [_RegexExecutorPolicy](#) : int { [_S_auto](#), [_S_alternate](#) }

Functions

- template<typename [_FwdIter](#), typename [_TraitsT](#) >
[__enable_if_contiguous_normal_iter](#)< [_FwdIter](#), [_TraitsT](#) > [__compile_nfa](#) ([_FwdIter](#) __first, [_FwdIter](#) __last,
const typename [_TraitsT](#)::locale_type & __loc, [regex_constants::syntax_option_type](#) __flags)
- template<typename [_FwdIter](#), typename [_TraitsT](#) >
[__disable_if_contiguous_normal_iter](#)< [_FwdIter](#), [_TraitsT](#) > [__compile_nfa](#) ([_FwdIter](#) __first, [_FwdIter](#) __last,
const typename [_TraitsT](#)::locale_type & __loc, [regex_constants::syntax_option_type](#) __flags)
- template<class [_Iterator](#) >
std::iterator_traits< [_Iterator](#) >::difference_type [__distance_fw](#) ([_Iterator](#) __first, [_Iterator](#) __last, [std::input_](#)↵
[iterator_tag](#))
- template<class [_Iterator](#) >
std::iterator_traits< [_Iterator](#) >::difference_type [__distance_fw](#) ([_Iterator](#) __first, [_Iterator](#) __last, [std::forward_](#)↵
[iterator_tag](#))
- template<class [_Iterator](#) >
std::iterator_traits< [_Iterator](#) >::difference_type [__distance_fw](#) ([_Iterator](#) __first, [_Iterator](#) __last)
- template<typename [_InputIterator](#), typename [_OutputIterator](#), typename [_Tp](#) >
[_OutputIterator](#) [__normalize](#) ([_InputIterator](#) __first, [_InputIterator](#) __last, [_OutputIterator](#) __result, const [_Tp](#) &↵
__factor)

- `template<typename _Bilter, typename _Alloc, typename _CharT, typename _Traits, _RegexExecutorPolicy __policy, bool __match_↵ mode>`
`bool __regex_algo_impl (_Bilter __s, _Bilter __e, match_results< _Bilter, _Alloc > &__m, const basic_regex< _CharT, _Traits> > &__re, regex_constants::match_flag_type __flags)`
- `template<typename _Tp >`
`bool _Power_of_2 (_Tp __x)`
- `template<typename _Value, bool _Cache_hash_code>`
`bool operator!= (const _Node_iterator_base< _Value, _Cache_hash_code > &__x, const _Node_iterator_↵ base< _Value, _Cache_hash_code > &__y) noexcept`
- `template<typename _Key, typename _Value, typename _ExtractKey, typename _H1, typename _H2, typename _Hash, bool __cache>`
`bool operator!= (const _Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache > &__x, const _Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache > &__y)`
- `template<typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const _Quoted_string< const _CharT *, _CharT > &__str)`
- `template<typename _CharT, typename _Traits, typename _String >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const _Quoted_string< _String, _CharT > &__str)`
- `template<typename _Value, bool _Cache_hash_code>`
`bool operator== (const _Node_iterator_base< _Value, _Cache_hash_code > &__x, const _Node_iterator_↵ base< _Value, _Cache_hash_code > &__y) noexcept`
- `template<typename _Key, typename _Value, typename _ExtractKey, typename _H1, typename _H2, typename _Hash, bool __cache>`
`bool operator== (const _Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache > &__x, const _Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache > &__y)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, const _↵ Quoted_string< basic_string< _CharT, _Traits, _Alloc > &, _CharT > &__str)`

Variables

- `static const _StateIdT _S_invalid_state_id`

4.13.1 Detailed Description

Implementation details not part of the namespace std interface.

4.13.2 Function Documentation

- 4.13.2.1 `template<typename _CharT, typename _Traits > std::basic_ostream< _CharT, _Traits> & std::__detail::operator<< (std::basic_ostream< _CharT, _Traits > & __os, const _Quoted_string< const _CharT *, _CharT > & __str)`

Insertor for quoted strings.

_GLIBCXX_RESOLVE_LIB_DEFECTS DR 2344 quoted()'s interaction with padding is unclear

Definition at line 75 of file quoted_string.h.

References `std::basic_ostringstream< _CharT, _Traits, _Alloc >::str()`.

4.13.2.2 `template<typename _CharT, typename _Traits, typename _String> std::basic_ostream<_CharT, _Traits>& std::__detail::operator<< (std::basic_ostream<_CharT, _Traits> & __os, const _Quoted_string<_String, _CharT> & __str)`

Inserter for quoted strings.

`_GLIBCXX_RESOLVE_LIB_DEFECTS` DR 2344 `quoted()`'s interaction with padding is unclear

Definition at line 99 of file `quoted_string.h`.

References `std::basic_ostringstream<_CharT, _Traits, _Alloc>::str()`.

4.13.2.3 `template<typename _CharT, typename _Traits, typename _Alloc> std::basic_istream<_CharT, _Traits>& std::__detail::operator>> (std::basic_istream<_CharT, _Traits> & __is, const _Quoted_string<_CharT, _Traits, _Alloc> & __str)`

Extractor for delimited strings. The left and right delimiters can be different.

Definition at line 121 of file `quoted_string.h`.

References `std::basic_ios<_CharT, _Traits>::clear()`, `std::ios_base::flags()`, `std::basic_ios<_CharT, _Traits>::good()`, `std::ios_base::setf()`, `std::skipws()`, and `std::basic_istream<_CharT, _Traits>::unget()`.

Referenced by `std::mersenne_twister_engine<_UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f>::discard()`, `std::operator!=()`, `std::subtract_with_carry_engine<_UIntType, __w, __s, __r>::operator()`, `std::discard_block_engine<_RandomNumberEngine, __p, __r>::operator()`, `std::shuffle_order_engine<_RandomNumberEngine, __k>::operator()`, `std::normal_distribution<result_type>::operator()`, `std::normal_distribution<_RealType>::operator()`, `std::gamma_distribution<_RealType>::operator()`, `std::binomial_distribution<_IntType>::operator()`, `std::negative_binomial_distribution<_IntType>::operator()`, `std::poisson_distribution<_IntType>::operator()`, `std::operator>>()`, and `std::linear_congruential_engine<_UIntType, __a, __c, __m>::seed()`.

4.14 `std::__parallel` Namespace Reference

Classes

- struct [`_CRandNumber`](#)

Functions

- `template<typename _Iter, typename _Tp, typename _Tag> _Tp __accumulate_switch (_Iter, _Iter, _Tp, _Tag)`
- `template<typename _Iter, typename _Tp, typename _IteratorTag> _Tp __accumulate_switch (_Iter __begin, _Iter __end, _Tp __init, _IteratorTag)`
- `template<typename _Iter, typename _Tp, typename _BinaryOper, typename _Tag> _Tp __accumulate_switch (_Iter, _Iter, _Tp, _BinaryOper, _Tag)`
- `template<typename _Iter, typename _Tp, typename _BinaryOperation, typename _IteratorTag> _Tp __accumulate_switch (_Iter __begin, _Iter __end, _Tp __init, _BinaryOperation __binary_op, _IteratorTag)`
- `template<typename _RAIter, typename _Tp, typename _BinaryOper> _Tp __accumulate_switch (_RAIter, _RAIter, _Tp, _BinaryOper, random_access_iterator_tag, __gnu_parallel::__Parallelism __parallelism=__gnu_parallel::parallel_unbalanced)`

- `template<typename __RAIter, typename _Tp, typename _BinaryOperation >`
`_Tp __accumulate_switch (__RAIter __begin, __RAIter __end, _Tp __init, _BinaryOperation __binary_op,`
`random_access_iterator_tag, __gnu_parallel::__Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper, typename _Tag1, typename _Tag2 >`
`_OIter __adjacent_difference_switch (_Iter, _Iter, _OIter, _BinaryOper, _Tag1, _Tag2)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >`
`_OIter __adjacent_difference_switch (_Iter, _Iter, _OIter, _BinaryOper, random_access_iterator_tag,`
`random_access_iterator_tag, __gnu_parallel::__Parallelism __parallelism=__gnu_parallel::parallel_unbalanced)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation, typename _IteratorTag1, typename _IteratorTag2 >`
`_OutputIterator __adjacent_difference_switch (_Iter __begin, _Iter __end, _OutputIterator __result, _Binary←`
`Operation __bin_op, _IteratorTag1, _IteratorTag2)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator __adjacent_difference_switch (_Iter __begin, _Iter __end, _OutputIterator __result, _Binary←`
`Operation __bin_op, random_access_iterator_tag, random_access_iterator_tag, __gnu_parallel::__Parallelism ←`
`__parallelism_tag)`
- `template<typename _Filter, typename _IterTag >`
`_Filter __adjacent_find_switch (_Filter, _Filter, _IterTag)`
- `template<typename _Filter, typename _BiPredicate, typename _IterTag >`
`_Filter __adjacent_find_switch (_Filter, _Filter, _BiPredicate, _IterTag)`
- `template<typename __RAIter, typename _BiPredicate >`
`__RAIter __adjacent_find_switch (__RAIter, __RAIter, _BiPredicate, random_access_iterator_tag)`
- `template<typename __RAIter >`
`__RAIter __adjacent_find_switch (__RAIter __begin, __RAIter __end, random_access_iterator_tag)`
- `template<typename _Filterator, typename _IteratorTag >`
`_Filterator __adjacent_find_switch (_Filterator __begin, _Filterator __end, _IteratorTag)`
- `template<typename _Filterator, typename _BinaryPredicate, typename _IteratorTag >`
`_Filterator __adjacent_find_switch (_Filterator __begin, _Filterator __end, _BinaryPredicate __pred, _Iterator←`
`Tag)`
- `template<typename __RAIter, typename _BinaryPredicate >`
`__RAIter __adjacent_find_switch (__RAIter __begin, __RAIter __end, _BinaryPredicate __pred, random_access←`
`__iterator_tag)`
- `template<typename _Iter, typename _Predicate, typename _IterTag >`
`iterator_traits< _Iter >::difference_type __count_if_switch (_Iter, _Iter, _Predicate, _IterTag)`
- `template<typename __RAIter, typename _Predicate >`
`iterator_traits< __RAIter >::difference_type __count_if_switch (__RAIter __begin, __RAIter __end, _Predicate ←`
`__pred, random_access_iterator_tag, __gnu_parallel::__Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Predicate, typename _IteratorTag >`
`iterator_traits< _Iter >::difference_type __count_if_switch (_Iter __begin, _Iter __end, _Predicate __pred,`
`_IteratorTag)`
- `template<typename _Iter, typename _Tp, typename _IterTag >`
`iterator_traits< _Iter >::difference_type __count_switch (_Iter, _Iter, const _Tp &, _IterTag)`
- `template<typename __RAIter, typename _Tp >`
`iterator_traits< __RAIter >::difference_type __count_switch (__RAIter __begin, __RAIter __end, const _Tp &←`
`value, random_access_iterator_tag, __gnu_parallel::__Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Tp, typename _IteratorTag >`
`iterator_traits< _Iter >::difference_type __count_switch (_Iter __begin, _Iter __end, const _Tp &__value, ←`
`_IteratorTag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IteratorTag1, typename _IteratorTag2 >`
`bool __equal_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _Predicate __pred,`
`_IteratorTag1, _IteratorTag2)`
- `template<typename __RAIter1, typename __RAIter2, typename _Predicate >`
`bool __equal_switch (__RAIter1 __begin1, __RAIter1 __end1, __RAIter2 __begin2, __RAIter2 __end2, _Predicate`
`__pred, random_access_iterator_tag, random_access_iterator_tag)`

- `template<typename _Iter, typename _Filter, typename _IterTag1, typename _IterTag2 >`
`_Iter __find_first_of_switch (_Iter, _Iter, _Filter, _Filter, _IterTag1, _IterTag2)`
- `template<typename _RAIter, typename _Filter, typename _BiPredicate, typename _IterTag >`
`_RAIter __find_first_of_switch (_RAIter, _RAIter, _Filter, _Filter, _BiPredicate, random_access_iterator_tag, ↵
_IterTag)`
- `template<typename _Iter, typename _Filter, typename _BiPredicate, typename _IterTag1, typename _IterTag2 >`
`_Iter __find_first_of_switch (_Iter, _Iter, _Filter, _Filter, _BiPredicate, _IterTag1, _IterTag2)`
- `template<typename _Iter, typename _FIterator, typename _IteratorTag1, typename _IteratorTag2 >`
`_Iter __find_first_of_switch (_Iter __begin1, _Iter __end1, _FIterator __begin2, _FIterator __end2, _Iterator↵
Tag1, _IteratorTag2)`
- `template<typename _RAIter, typename _FIterator, typename _BinaryPredicate, typename _IteratorTag >`
`_RAIter __find_first_of_switch (_RAIter __begin1, _RAIter __end1, _FIterator __begin2, _FIterator __end2,
_BinaryPredicate __comp, random_access_iterator_tag, _IteratorTag)`
- `template<typename _Iter, typename _FIterator, typename _BinaryPredicate, typename _IteratorTag1, typename _IteratorTag2 >`
`_Iter __find_first_of_switch (_Iter __begin1, _Iter __end1, _FIterator __begin2, _FIterator __end2, _Binary↵
Predicate __comp, _IteratorTag1, _IteratorTag2)`
- `template<typename _Iter, typename _Predicate, typename _IterTag >`
`_Iter __find_if_switch (_Iter, _Iter, _Predicate, _IterTag)`
- `template<typename _Iter, typename _Predicate, typename _IteratorTag >`
`_Iter __find_if_switch (_Iter __begin, _Iter __end, _Predicate __pred, _IteratorTag)`
- `template<typename _RAIter, typename _Predicate >`
`_RAIter __find_if_switch (_RAIter __begin, _RAIter __end, _Predicate __pred, random_access_iterator_tag)`
- `template<typename _Iter, typename _Tp, typename _IteratorTag >`
`_Iter __find_switch (_Iter __begin, _Iter __end, const _Tp &__val, _IteratorTag)`
- `template<typename _RAIter, typename _Tp >`
`_RAIter __find_switch (_RAIter __begin, _RAIter __end, const _Tp &__val, random_access_iterator_tag)`
- `template<typename _Iter, typename _Tp, typename _IterTag >`
`_Iter __find_switch (_Iter, _Iter, const _Tp &, _IterTag)`
- `template<typename _Iter, typename _Function, typename _IteratorTag >`
`_Function __for_each_switch (_Iter __begin, _Iter __end, _Function __f, _IteratorTag)`
- `template<typename _RAIter, typename _Function >`
`_Function __for_each_switch (_RAIter __begin, _RAIter __end, _Function __f, random_access_iterator_tag,
__gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Function, typename _IterTag >`
`_Function __for_each_switch (_Iter, _Iter, _Function, _IterTag)`
- `template<typename _OIter, typename _Size, typename _Generator, typename _IterTag >`
`_OIter __generate_n_switch (_OIter, _Size, _Generator, _IterTag)`
- `template<typename _OutputIterator, typename _Size, typename _Generator, typename _IterTag >`
`_OutputIterator __generate_n_switch (_OutputIterator __begin, _Size __n, _Generator __gen, _IteratorTag)`
- `template<typename _RAIter, typename _Size, typename _Generator >`
`_RAIter __generate_n_switch (_RAIter __begin, _Size __n, _Generator __gen, random_access_iterator_tag,
__gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filter, typename _Generator, typename _IterTag >`
`void __generate_switch (_Filter, _Filter, _Generator, _IterTag)`
- `template<typename _FIterator, typename _Generator, typename _IteratorTag >`
`void __generate_switch (_FIterator __begin, _FIterator __end, _Generator __gen, _IteratorTag)`
- `template<typename _RAIter, typename _Generator >`
`void __generate_switch (_RAIter __begin, _RAIter __end, _Generator __gen, random_access_iterator_tag, ↵
__gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _RAIter1, typename _RAIter2, typename _Tp, typename BinaryFunction1, typename BinaryFunction2 >`
`_Tp __inner_product_switch (_RAIter1, _RAIter1, _RAIter2, _Tp, BinaryFunction1, BinaryFunction2, random↵
_access_iterator_tag, random_access_iterator_tag, __gnu_parallel::Parallelism=__gnu_parallel::parallel↵
unbalanced)`

- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2, typename _Tag1, typename _Tag2 >`
`_Tp __inner_product_switch (_Iter1, _Iter1, _Iter2, _Tp, _BinaryFunction1, _BinaryFunction2, _Tag1, _Tag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 >`
`_Tp __inner_product_switch (_RAIter1 __first1, _RAIter1 __last1, _RAIter2 __first2, _Tp __init, _BinaryFunction1 __binary_op1, _BinaryFunction2 __binary_op2, random_access_iterator_tag, random_access_iterator_tag, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2, typename _IterTag1, typename _IterTag2 >`
`_Tp __inner_product_switch (_Iter1 __first1, _Iter1 __last1, _Iter2 __first2, _Tp __init, _BinaryFunction1 __binary_op1, _BinaryFunction2 __binary_op2, _IterTag1, _IterTag2)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IterTag1, typename _IterTag2 >`
`bool __lexicographical_compare_switch (_Iter1, _Iter1, _Iter2, _Iter2, _Predicate, _IterTag1, _IterTag2)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IterTag1, typename _IterTag2 >`
`bool __lexicographical_compare_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _Predicate __pred, _IterTag1, _IterTag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _Predicate >`
`bool __lexicographical_compare_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Filter, typename _Compare, typename _IterTag >`
`_Filter __max_element_switch (_Filter, _Filter, _Compare, _IterTag)`
- `template<typename _Filterator, typename _Compare, typename _IterTag >`
`_Filterator __max_element_switch (_Filterator __begin, _Filterator __end, _Compare __comp, _IterTag)`
- `template<typename _RAIter, typename _Compare >`
`_RAIter __max_element_switch (_RAIter __begin, _RAIter __end, _Compare __comp, random_access_iterator_tag, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare, typename _IterTag1, typename _IterTag2, typename _IterTag3 >`
`_OIter __merge_switch (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare, _IterTag1, _IterTag2, _IterTag3)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`_OIter __merge_switch (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare, random_access_iterator_tag, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Compare, typename _IterTag1, typename _IterTag2, typename _IterTag3 >`
`_OutputIterator __merge_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __result, _Compare __comp, _IterTag1, _IterTag2, _IterTag3)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Compare >`
`_OutputIterator __merge_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __result, _Compare __comp, random_access_iterator_tag, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Filter, typename _Compare, typename _IterTag >`
`_Filter __min_element_switch (_Filter, _Filter, _Compare, _IterTag)`
- `template<typename _Filterator, typename _Compare, typename _IterTag >`
`_Filterator __min_element_switch (_Filterator __begin, _Filterator __end, _Compare __comp, _IterTag)`
- `template<typename _RAIter, typename _Compare >`
`_RAIter __min_element_switch (_RAIter __begin, _RAIter __end, _Compare __comp, random_access_iterator_tag, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IterTag1, typename _IterTag2 >`
`pair< _Iter1, _Iter2 > __mismatch_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Predicate __pred, _IterTag1, _IterTag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _Predicate >`
`pair< _RAIter1, _RAIter2 > __mismatch_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)`

- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IteratorTag1, typename _IteratorTag2 >`
`pair< _Iter1, _Iter2 > __mismatch_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2,`
`_Predicate __pred, _IteratorTag1, _IteratorTag2)`
- `template<typename _RAlter, typename _RAlter2, typename _Predicate >`
`pair< _RAlter1, _RAlter2 > __mismatch_switch (_RAlter1 __begin1, _RAlter1 __end1, _RAlter2 __begin2,`
`_RAlter2 __end2, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IterTag1, typename _IterTag2 >`
`pair< _Iter1, _Iter2 > __mismatch_switch (_Iter1, _Iter1, _Iter2, _Predicate, _IterTag1, _IterTag2)`
- `template<typename _Iter, typename _Olter, typename _BinaryOper, typename _Tag1, typename _Tag2 >`
`_Olter __partial_sum_switch (_Iter, _Iter, _Olter, _BinaryOper, _Tag1, _Tag2)`
- `template<typename _Iter, typename _Olter, typename _BinaryOper >`
`_Olter __partial_sum_switch (_Iter, _Iter, _Olter, _BinaryOper, random_access_iterator_tag, random_↵`
`access_iterator_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation, typename _IteratorTag1, typename _IteratorTag2 >`
`_OutputIterator __partial_sum_switch (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation`
`__bin_op, _IteratorTag1, _IteratorTag2)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator __partial_sum_switch (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation`
`__bin_op, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Filter, typename _Predicate, typename _IterTag >`
`_Filter __partition_switch (_Filter, _Filter, _Predicate, _IterTag)`
- `template<typename _Filterator, typename _Predicate, typename _IteratorTag >`
`_Filterator __partition_switch (_Filterator __begin, _Filterator __end, _Predicate __pred, _IteratorTag)`
- `template<typename _RAlter, typename _Predicate >`
`_RAlter __partition_switch (_RAlter __begin, _RAlter __end, _Predicate __pred, random_access_iterator_tag)`
- `template<typename _Filter, typename _Predicate, typename _Tp, typename _IterTag >`
`void __replace_if_switch (_Filter, _Filter, _Predicate, const _Tp &, _IterTag)`
- `template<typename _Filterator, typename _Predicate, typename _Tp, typename _IteratorTag >`
`void __replace_if_switch (_Filterator __begin, _Filterator __end, _Predicate __pred, const _Tp & __new_value,`
`_IteratorTag)`
- `template<typename _RAlter, typename _Predicate, typename _Tp >`
`void __replace_if_switch (_RAlter __begin, _RAlter __end, _Predicate __pred, const _Tp & __new_value,`
`random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filter, typename _Tp, typename _IterTag >`
`void __replace_switch (_Filter, _Filter, const _Tp &, const _Tp &, _IterTag)`
- `template<typename _Filterator, typename _Tp, typename _IteratorTag >`
`void __replace_switch (_Filterator __begin, _Filterator __end, const _Tp & __old_value, const _Tp & __new_value,`
`_IteratorTag)`
- `template<typename _RAlter, typename _Tp >`
`void __replace_switch (_RAlter __begin, _RAlter __end, const _Tp & __old_value, const _Tp & __new_value,`
`random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _RAlter, typename _Integer, typename _Tp, typename _BiPredicate >`
`_RAlter __search_n_switch (_RAlter, _RAlter, _Integer, const _Tp &, _BiPredicate, random_access_iterator_↵`
`tag)`
- `template<typename _Filter, typename _Integer, typename _Tp, typename _BiPredicate, typename _IterTag >`
`_Filter __search_n_switch (_Filter, _Filter, _Integer, const _Tp &, _BiPredicate, _IterTag)`
- `template<typename _RAlter, typename _Integer, typename _Tp, typename _BinaryPredicate >`
`_RAlter __search_n_switch (_RAlter __begin, _RAlter __end, _Integer __count, const _Tp & __val, _Binary↵`
`Predicate __binary_pred, random_access_iterator_tag)`
- `template<typename _Filterator, typename _Integer, typename _Tp, typename _BinaryPredicate, typename _IteratorTag >`
`_Filterator __search_n_switch (_Filterator __begin, _Filterator __end, _Integer __count, const _Tp & __val, ↵`
`BinaryPredicate __binary_pred, _IteratorTag)`

- `template<typename _Filter1, typename _Filter2, typename _IterTag1, typename _IterTag2 >
_Filter1 __search_switch (_Filter1, _Filter1, _Filter2, _Filter2, _IterTag1, _IterTag2)`
- `template<typename _RAlter1, typename _RAlter2, typename _BiPredicate >
_RAlter1 __search_switch (_RAlter1, _RAlter1, _RAlter2, _RAlter2, _BiPredicate, random_access_iterator_tag,
random_access_iterator_tag)`
- `template<typename _Filter1, typename _Filter2, typename _BiPredicate, typename _IterTag1, typename _IterTag2 >
_Filter1 __search_switch (_Filter1, _Filter1, _Filter2, _Filter2, _BiPredicate, _IterTag1, _IterTag2)`
- `template<typename _RAlter1, typename _RAlter2 >
_RAlter1 __search_switch (_RAlter1 __begin1, _RAlter1 __end1, _RAlter2 __begin2, _RAlter2 __end2,
random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _FIterator1, typename _FIterator2, typename _IteratorTag1, typename _IteratorTag2 >
_FIterator1 __search_switch (_FIterator1 __begin1, _FIterator1 __end1, _FIterator2 __begin2, _FIterator2 __↵
end2, _IteratorTag1, _IteratorTag2)`
- `template<typename _RAlter1, typename _RAlter2, typename _BinaryPredicate >
_RAlter1 __search_switch (_RAlter1 __begin1, _RAlter1 __end1, _RAlter2 __begin2, _RAlter2 __end2, __↵
BinaryPredicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _FIterator1, typename _FIterator2, typename _BinaryPredicate, typename _IteratorTag1, typename _IteratorTag2 >
_FIterator1 __search_switch (_FIterator1 __begin1, _FIterator1 __end1, _FIterator2 __begin2, _FIterator2 __↵
end2, _BinaryPredicate __pred, _IteratorTag1, _IteratorTag2)`
- `template<typename _IIter1, typename _IIter2, typename _Predicate, typename _OutputIterator, typename _IteratorTag1, typename __↵
IteratorTag2, typename _IteratorTag3 >
_OutputIterator __set_difference_switch (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 __end2,
_OutputIterator __result, _Predicate __pred, _IteratorTag1, _IteratorTag2, _IteratorTag3)`
- `template<typename _RAlter1, typename _RAlter2, typename _Output_RAlter, typename _Predicate >
_Output_RAlter __set_difference_switch (_RAlter1 __begin1, _RAlter1 __end1, _RAlter2 __begin2, _RAlter2
__end2, _Output_RAlter __result, _Predicate __pred, random_access_iterator_tag, random_access_iterator__↵
tag, random_access_iterator_tag)`
- `template<typename _IIter1, typename _IIter2, typename _Predicate, typename _OIter, typename _IterTag1, typename _IterTag2, type-
name _IterTag3 >
_OIter __set_difference_switch (_IIter1, _IIter1, _IIter2, _IIter2, _OIter, _Predicate, _IterTag1, _IterTag2, _Iter__↵
Tag3)`
- `template<typename _IIter1, typename _IIter2, typename _Predicate, typename _OutputIterator, typename _IteratorTag1, typename __↵
IteratorTag2, typename _IteratorTag3 >
_OutputIterator __set_intersection_switch (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 __end2,
_OutputIterator __result, _Predicate __pred, _IteratorTag1, _IteratorTag2, _IteratorTag3)`
- `template<typename _RAlter1, typename _RAlter2, typename _Output_RAlter, typename _Predicate >
_Output_RAlter __set_intersection_switch (_RAlter1 __begin1, _RAlter1 __end1, _RAlter2 __begin2, _RAlter2
__end2, _Output_RAlter __result, _Predicate __pred, random_access_iterator_tag, random_access_iterator__↵
tag, random_access_iterator_tag)`
- `template<typename _IIter1, typename _IIter2, typename _Predicate, typename _OIter, typename _IterTag1, typename _IterTag2, type-
name _IterTag3 >
_OIter __set_intersection_switch (_IIter1, _IIter1, _IIter2, _IIter2, _OIter, _Predicate, _IterTag1, _IterTag2, __↵
IterTag3)`
- `template<typename _IIter1, typename _IIter2, typename _Predicate, typename _OutputIterator, typename _IteratorTag1, typename __↵
IteratorTag2, typename _IteratorTag3 >
_OutputIterator __set_symmetric_difference_switch (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2
__end2, _OutputIterator __result, _Predicate __pred, _IteratorTag1, _IteratorTag2, _IteratorTag3)`
- `template<typename _RAlter1, typename _RAlter2, typename _Output_RAlter, typename _Predicate >
_Output_RAlter __set_symmetric_difference_switch (_RAlter1 __begin1, _RAlter1 __end1, _RAlter2 __↵
begin2, _RAlter2 __end2, _Output_RAlter __result, _Predicate __pred, random_access_iterator_tag, random__↵
_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _IIter1, typename _IIter2, typename _Predicate, typename _OIter, typename _IterTag1, typename _IterTag2, type-
name _IterTag3 >`

- `_OIter __set_symmetric_difference_switch (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate, _IterTag1, _IterTag2, _IterTag3)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _OutputIterator, typename _IteratorTag1, typename _IteratorTag2, typename _IteratorTag3 >`
`_OutputIterator __set_union_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __result, _Predicate __pred, _IteratorTag1, _IteratorTag2, _IteratorTag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _Output_RAlter, typename _Predicate >`
`_Output_RAlter __set_union_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, _Output_RAlter __result, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _OIter, typename _IterTag1, typename _IterTag2, typename _IterTag3 >`
`_OIter __set_union_switch (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate, _IterTag1, _IterTag2, _IterTag3)`
- `template<typename _Iter, typename _OIter, typename _UnaryOperation, typename _IterTag1, typename _IterTag2 >`
`_OIter __transform1_switch (_Iter, _Iter, _OIter, _UnaryOperation, _IterTag1, _IterTag2)`
- `template<typename _RAIter, typename _RAOIter, typename _UnaryOperation >`
`_RAOIter __transform1_switch (_RAIter, _RAIter, _RAOIter, _UnaryOperation, random_access_iterator_tag, random_access_iterator_tag, __gnu_parallel::Parallelism __parallelism=__gnu_parallel::parallel_balanced)`
- `template<typename _RAIter1, typename _RAIter2, typename _UnaryOperation >`
`_RAIter2 __transform1_switch (_RAIter1 __begin, _RAIter1 __end, _RAIter2 __result, _UnaryOperation __unary_op, random_access_iterator_tag, random_access_iterator_tag, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _RAIter1, typename _RAIter2, typename _UnaryOperation, typename _IteratorTag1, typename _IteratorTag2 >`
`_RAIter2 __transform1_switch (_RAIter1 __begin, _RAIter1 __end, _RAIter2 __result, _UnaryOperation __unary_op, _IteratorTag1, _IteratorTag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _RAIter3, typename _BiOperation >`
`_RAIter3 __transform2_switch (_RAIter1, _RAIter1, _RAIter2, _RAIter3, _BiOperation, random_access_iterator_tag, random_access_iterator_tag, random_access_iterator_tag, __gnu_parallel::Parallelism __parallelism=__gnu_parallel::parallel_balanced)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _BiOperation, typename _Tag1, typename _Tag2, typename _Tag3 >`
`_OIter __transform2_switch (_Iter1, _Iter1, _Iter2, _OIter, _BiOperation, _Tag1, _Tag2, _Tag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _RAIter3, typename _BinaryOperation >`
`_RAIter3 __transform2_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter3 __result, _BinaryOperation __binary_op, random_access_iterator_tag, random_access_iterator_tag, random_access_iterator_tag, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _BinaryOperation, typename _Tag1, typename _Tag2, typename _Tag3 >`
`_OutputIterator __transform2_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _OutputIterator __result, _BinaryOperation __binary_op, _Tag1, _Tag2, _Tag3)`
- `template<typename _Iter, typename _OutputIterator, typename _Predicate, typename _IteratorTag1, typename _IteratorTag2 >`
`_OutputIterator __unique_copy_switch (_Iter __begin, _Iter __last, _OutputIterator __out, _Predicate __pred, _IteratorTag1, _IteratorTag2)`
- `template<typename _RAIter, typename RandomAccessOutputIterator, typename _Predicate >`
`RandomAccessOutputIterator __unique_copy_switch (_RAIter __begin, _RAIter __last, RandomAccessOutputIterator __out, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter, typename _OIter, typename _Predicate, typename _IterTag1, typename _IterTag2 >`
`_OIter __unique_copy_switch (_Iter, _Iter, _OIter, _Predicate, _IterTag1, _IterTag2)`
- `template<typename _RAIter, typename _RandomAccess_OIter, typename _Predicate >`
`_RandomAccess_OIter __unique_copy_switch (_RAIter, _RAIter, _RandomAccess_OIter, _Predicate, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter, typename _Tp >`
`_Tp __accumulate (_Iter, _Iter, _Tp)`

- `template<typename _Iter, typename _Tp >`
`_Tp accumulate (_Iter, _Iter, _Tp, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Tp >`
`_Tp accumulate (_Iter, _Iter, _Tp, __gnu_parallel::Parallelism)`
- `template<typename _Iter, typename _Tp, typename _BinaryOper >`
`_Tp accumulate (_Iter, _Iter, _Tp, _BinaryOper)`
- `template<typename _Iter, typename _Tp, typename _BinaryOperation >`
`_Tp accumulate (_Iter __begin, _Iter __end, _Tp __init, _BinaryOperation __binary_op, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Tp, typename _BinaryOper >`
`_Tp accumulate (_Iter, _Iter, _Tp, _BinaryOper, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Tp, typename _BinaryOper >`
`_Tp accumulate (_Iter, _Iter, _Tp, _BinaryOper, __gnu_parallel::Parallelism)`
- `template<typename _Iter, typename _Tp, typename _BinaryOperation >`
`_Tp accumulate (_Iter __begin, _Iter __end, _Tp __init, _BinaryOperation __binary_op, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Tp, typename _BinaryOperation >`
`_Tp accumulate (_Iter __begin, _Iter __end, _Tp __init, _BinaryOperation __binary_op)`
- `template<typename _Iter, typename _OIter >`
`_OIter adjacent_difference (_Iter, _Iter, _OIter)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >`
`_OIter adjacent_difference (_Iter, _Iter, _OIter, _BinaryOper)`
- `template<typename _Iter, typename _OIter >`
`_OIter adjacent_difference (_Iter, _Iter, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >`
`_OIter adjacent_difference (_Iter, _Iter, _OIter, _BinaryOper, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter >`
`_OIter adjacent_difference (_Iter, _Iter, _OIter, __gnu_parallel::Parallelism)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >`
`_OIter adjacent_difference (_Iter, _Iter, _OIter, _BinaryOper, __gnu_parallel::Parallelism)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __bin_op, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __binary_op, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __binary_op)`
- `template<typename _Filter >`
`_Filter adjacent_find (_Filter, _Filter)`
- `template<typename _Filter >`
`_Filter adjacent_find (_Filter, _Filter, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _BiPredicate >`
`_Filter adjacent_find (_Filter, _Filter, _BiPredicate)`

- `template<typename _Filter, typename _BiPredicate >`
`_Filter adjacent_find (_Filter, _Filter, _BiPredicate, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator >`
`_FIterator adjacent_find (_FIterator __begin, _FIterator __end, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator, typename _BinaryPredicate >`
`_FIterator adjacent_find (_FIterator __begin, _FIterator __end, _BinaryPredicate __binary_pred, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator >`
`_FIterator adjacent_find (_FIterator __begin, _FIterator __end)`
- `template<typename _FIterator, typename _BinaryPredicate >`
`_FIterator adjacent_find (_FIterator __begin, _FIterator __end, _BinaryPredicate __pred)`
- `template<typename _Iter, typename _Tp >`
`iterator_traits< _Iter >::difference_type count (_Iter __begin, _Iter __end, const _Tp &__value, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Tp >`
`iterator_traits< _Iter >::difference_type count (_Iter __begin, _Iter __end, const _Tp &__value, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Tp >`
`iterator_traits< _Iter >::difference_type count (_Iter __begin, _Iter __end, const _Tp &__value)`
- `template<typename _Iter, typename _Predicate >`
`iterator_traits< _Iter >::difference_type count_if (_Iter __begin, _Iter __end, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Predicate >`
`iterator_traits< _Iter >::difference_type count_if (_Iter __begin, _Iter __end, _Predicate __pred, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Predicate >`
`iterator_traits< _Iter >::difference_type count_if (_Iter __begin, _Iter __end, _Predicate __pred)`
- `template<typename _Iter1, typename _Iter2 >`
`bool equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`bool equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`bool equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`bool equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Predicate __pred)`
- `template<typename _Iter1, typename _Iter2 >`
`bool equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _BinaryPredicate >`
`bool equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _BinaryPredicate __binary_pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`bool equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2)`
- `template<typename _Iter1, typename _Iter2, typename _BinaryPredicate >`
`bool equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _BinaryPredicate __binary_pred)`
- `template<typename _Iter, typename _Tp >`
`_Iter find (_Iter __begin, _Iter __end, const _Tp &__val, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Tp >`
`_Iter find (_Iter __begin, _Iter __end, const _Tp &__val)`
- `template<typename _Iter, typename _Filter >`
`_Iter find_first_of (_Iter, _Iter, _Filter, _Filter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Filter, typename _BiPredicate >`
`_Iter find_first_of (_Iter, _Iter, _Filter, _Filter, _BiPredicate, __gnu_parallel::sequential_tag)`

- `template<typename _Iter, typename _Filter, typename _BiPredicate >`
`_Iter find_first_of (_Iter, _Iter, _Filter, _Filter, _BiPredicate)`
- `template<typename _Iter, typename _Filter >`
`_Iter find_first_of (_Iter, _Iter, _Filter, _Filter)`
- `template<typename _Iter, typename _FIterator >`
`_Iter find_first_of (_Iter __begin1, _Iter __end1, _FIterator __begin2, _FIterator __end2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _FIterator, typename _BinaryPredicate >`
`_Iter find_first_of (_Iter __begin1, _Iter __end1, _FIterator __begin2, _FIterator __end2, _BinaryPredicate __comp, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _FIterator, typename _BinaryPredicate >`
`_Iter find_first_of (_Iter __begin1, _Iter __end1, _FIterator __begin2, _FIterator __end2, _BinaryPredicate __comp)`
- `template<typename _Iter, typename _FIterator >`
`_Iter find_first_of (_Iter __begin1, _Iter __end1, _FIterator __begin2, _FIterator __end2)`
- `template<typename _Iter, typename _Predicate >`
`_Iter find_if (_Iter __begin, _Iter __end, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Predicate >`
`_Iter find_if (_Iter __begin, _Iter __end, _Predicate __pred)`
- `template<typename _Iter, typename _Function >`
`_Function for_each (_Iter __begin, _Iter __end, _Function __f, __gnu_parallel::sequential_tag)`
- `template<typename _Iterator, typename _Function >`
`_Function for_each (_Iterator __begin, _Iterator __end, _Function __f, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iterator, typename _Function >`
`_Function for_each (_Iterator __begin, _Iterator __end, _Function __f)`
- `template<typename _Iter, typename _Function >`
`_Function for_each (_Iter, _Iter, _Function)`
- `template<typename _Filter, typename _Generator >`
`void generate (_Filter, _Filter, _Generator)`
- `template<typename _Filter, typename _Generator >`
`void generate (_Filter, _Filter, _Generator, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Generator >`
`void generate (_Filter, _Filter, _Generator, __gnu_parallel::Parallelism)`
- `template<typename _FIterator, typename _Generator >`
`void generate (_FIterator __begin, _FIterator __end, _Generator __gen, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator, typename _Generator >`
`void generate (_FIterator __begin, _FIterator __end, _Generator __gen, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _FIterator, typename _Generator >`
`void generate (_FIterator __begin, _FIterator __end, _Generator __gen)`
- `template<typename _OIter, typename _Size, typename _Generator >`
`_OIter generate_n (_OIter, _Size, _Generator)`
- `template<typename _OIter, typename _Size, typename _Generator >`
`_OIter generate_n (_OIter, _Size, _Generator, __gnu_parallel::sequential_tag)`
- `template<typename _OIter, typename _Size, typename _Generator >`
`_OIter generate_n (_OIter, _Size, _Generator, __gnu_parallel::Parallelism)`
- `template<typename _OutputIterator, typename _Size, typename _Generator >`
`_OutputIterator generate_n (_OutputIterator __begin, _Size __n, _Generator __gen, __gnu_parallel::sequential_tag)`
- `template<typename _OutputIterator, typename _Size, typename _Generator >`
`_OutputIterator generate_n (_OutputIterator __begin, _Size __n, _Generator __gen, __gnu_parallel::Parallelism __parallelism_tag)`

- `template<typename _OutputIterator, typename _Size, typename _Generator >`
`_OutputIterator generate_n (_OutputIterator __begin, _Size __n, _Generator __gen)`
- `template<typename _Iter1, typename _Iter2, typename _Tp >`
`_Tp inner_product (_Iter1, _Iter1, _Iter2, _Tp)`
- `template<typename _Iter1, typename _Iter2, typename _Tp >`
`_Tp inner_product (_Iter1, _Iter1, _Iter2, _Tp, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Tp >`
`_Tp inner_product (_Iter1, _Iter1, _Iter2, _Tp, __gnu_parallel::Parallelism)`
- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 >`
`_Tp inner_product (_Iter1, _Iter1, _Iter2, _Tp, _BinaryFunction1, _BinaryFunction2)`
- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 >`
`_Tp inner_product (_Iter1, _Iter1, _Iter2, _Tp, _BinaryFunction1, _BinaryFunction2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Tp, typename BinaryFunction1, typename BinaryFunction2 >`
`_Tp inner_product (_Iter1, _Iter1, _Iter2, _Tp, BinaryFunction1, BinaryFunction2, __gnu_parallel::Parallelism)`
- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 >`
`_Tp inner_product (_Iter1 __first1, _Iter1 __last1, _Iter2 __first2, _Tp __init, _BinaryFunction1 __binary_op1, _BinaryFunction2 __binary_op2, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`bool lexicographical_compare (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`bool lexicographical_compare (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`bool lexicographical_compare (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`bool lexicographical_compare (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _Predicate __pred)`
- `template<typename _Filter >`
`_Filter max_element (_Filter, _Filter)`
- `template<typename _Filter >`
`_Filter max_element (_Filter, _Filter, __gnu_parallel::sequential_tag)`
- `template<typename _Filter >`
`_Filter max_element (_Filter, _Filter, __gnu_parallel::Parallelism)`
- `template<typename _Filter, typename _Compare >`
`_Filter max_element (_Filter, _Filter, _Compare)`
- `template<typename _Filter, typename _Compare >`
`_Filter max_element (_Filter, _Filter, _Compare, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Compare >`
`_Filter max_element (_Filter, _Filter, _Compare, __gnu_parallel::Parallelism)`
- `template<typename _Filterator >`
`_Filterator max_element (_Filterator __begin, _Filterator __end, __gnu_parallel::sequential_tag)`
- `template<typename _Filterator, typename _Compare >`
`_Filterator max_element (_Filterator __begin, _Filterator __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _Filterator >`
`_Filterator max_element (_Filterator __begin, _Filterator __end, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filterator >`
`_Filterator max_element (_Filterator __begin, _Filterator __end)`

- `template<typename _FIterator, typename _Compare >`
`_FIterator max_element (_FIterator __begin, _FIterator __end, _Compare __comp, __gnu_parallel::Parallelism`
`__parallelism_tag)`
- `template<typename _FIterator, typename _Compare >`
`_FIterator max_element (_FIterator __begin, _FIterator __end, _Compare __comp)`
- `template<typename _IIter1, typename _IIter2, typename _OIter >`
`_OIter merge (_IIter1, _IIter1, _IIter2, _IIter2, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _OIter, typename _Compare >`
`_OIter merge (_IIter1, _IIter1, _IIter2, _IIter2, _OIter, _Compare, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _OIter, typename _Compare >`
`_OIter merge (_IIter1, _IIter1, _IIter2, _IIter2, _OIter, _Compare)`
- `template<typename _IIter1, typename _IIter2, typename _OIter >`
`_OIter merge (_IIter1, _IIter1, _IIter2, _IIter2, _OIter)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator >`
`_OutputIterator merge (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 __end2, _OutputIterator __↵`
`result, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator, typename _Compare >`
`_OutputIterator merge (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 __end2, _OutputIterator __↵`
`result, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator, typename _Compare >`
`_OutputIterator merge (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 __end2, _OutputIterator __↵`
`result, _Compare __comp)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator >`
`_OutputIterator merge (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 __end2, _OutputIterator __↵`
`result)`
- `template<typename _Filter >`
`_Filter min_element (_Filter, _Filter)`
- `template<typename _Filter >`
`_Filter min_element (_Filter, _Filter, __gnu_parallel::sequential_tag)`
- `template<typename _Filter >`
`_Filter min_element (_Filter, _Filter, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filter, typename _Compare >`
`_Filter min_element (_Filter, _Filter, _Compare)`
- `template<typename _Filter, typename _Compare >`
`_Filter min_element (_Filter, _Filter, _Compare, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Compare >`
`_Filter min_element (_Filter, _Filter, _Compare, __gnu_parallel::Parallelism)`
- `template<typename _FIterator >`
`_FIterator min_element (_FIterator __begin, _FIterator __end, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator, typename _Compare >`
`_FIterator min_element (_FIterator __begin, _FIterator __end, _Compare __comp, __gnu_parallel::sequential↵`
`_tag)`
- `template<typename _FIterator >`
`_FIterator min_element (_FIterator __begin, _FIterator __end, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _FIterator >`
`_FIterator min_element (_FIterator __begin, _FIterator __end)`
- `template<typename _FIterator, typename _Compare >`
`_FIterator min_element (_FIterator __begin, _FIterator __end, _Compare __comp, __gnu_parallel::Parallelism`
`__parallelism_tag)`
- `template<typename _FIterator, typename _Compare >`
`_FIterator min_element (_FIterator __begin, _FIterator __end, _Compare __comp)`

- `template<typename _Iter1, typename _Iter2 >`
`pair< _Iter1, _Iter2 > mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`pair< _Iter1, _Iter2 > mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`pair< _Iter1, _Iter2 > mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`pair< _Iter1, _Iter2 > mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Predicate __pred)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`pair< _InputIterator1, _InputIterator2 > mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, __gnu_parallel::sequential_tag)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`pair< _InputIterator1, _InputIterator2 > mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _BinaryPredicate __binary_pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`pair< _Iter1, _Iter2 > mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`pair< _InputIterator1, _InputIterator2 > mismatch (_InputIterator1 __begin1, _InputIterator1 __end1, _InputIterator2 __begin2, _InputIterator2 __end2, _BinaryPredicate __binary_pred)`
- `template<typename _RAIter >`
`void nth_element (_RAIter __begin, _RAIter __nth, _RAIter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare >`
`void nth_element (_RAIter __begin, _RAIter __nth, _RAIter __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare >`
`void nth_element (_RAIter __begin, _RAIter __nth, _RAIter __end, _Compare __comp)`
- `template<typename _RAIter >`
`void nth_element (_RAIter __begin, _RAIter __nth, _RAIter __end)`
- `template<typename _RAIter, typename _Compare >`
`void partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter >`
`void partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare >`
`void partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end, _Compare __comp)`
- `template<typename _RAIter >`
`void partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end)`
- `template<typename _Iter, typename _OIter >`
`_OIter partial_sum (_Iter, _Iter, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >`
`_OIter partial_sum (_Iter, _Iter, _OIter, _BinaryOper, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter >`
`_OIter partial_sum (_Iter, _Iter, _OIter __result)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >`
`_OIter partial_sum (_Iter, _Iter, _OIter, _BinaryOper)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator partial_sum (_Iter __begin, _Iter __end, _OutputIterator __result, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator partial_sum (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __bin_op, __gnu_parallel::sequential_tag)`

- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator partial_sum (_Iter __begin, _Iter __end, _OutputIterator __result)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator partial_sum (_Iter __begin, _Iter __end, _OutputIterator __result, _BinaryOperation __binary←`
`__op)`
- `template<typename _Filter, typename _Predicate >`
`_Filter partition (_Filter, _Filter, _Predicate, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Predicate >`
`_Filter partition (_Filter, _Filter, _Predicate)`
- `template<typename _Filterator, typename _Predicate >`
`_Filterator partition (_Filterator __begin, _Filterator __end, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Filterator, typename _Predicate >`
`_Filterator partition (_Filterator __begin, _Filterator __end, _Predicate __pred)`
- `template<typename _RAlter >`
`void random_shuffle (_RAlter __begin, _RAlter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAlter, typename _RandomNumberGenerator >`
`void random_shuffle (_RAlter __begin, _RAlter __end, _RandomNumberGenerator &__rand, __gnu_parallel::sequential_tag)`
- `template<typename _RAlter >`
`void random_shuffle (_RAlter __begin, _RAlter __end)`
- `template<typename _RAlter, typename _RandomNumberGenerator >`
`void random_shuffle (_RAlter __begin, _RAlter __end, _RandomNumberGenerator &&__rand)`
- `template<typename _Filter, typename _Tp >`
`void replace (_Filter, _Filter, const _Tp &, const _Tp &)`
- `template<typename _Filter, typename _Tp >`
`void replace (_Filter, _Filter, const _Tp &, const _Tp &, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Tp >`
`void replace (_Filter, _Filter, const _Tp &, const _Tp &, __gnu_parallel::Parallelism)`
- `template<typename _Filterator, typename _Tp >`
`void replace (_Filterator __begin, _Filterator __end, const _Tp &__old_value, const _Tp &__new_value, __gnu_parallel::sequential_tag)`
- `template<typename _Filterator, typename _Tp >`
`void replace (_Filterator __begin, _Filterator __end, const _Tp &__old_value, const _Tp &__new_value, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filterator, typename _Tp >`
`void replace (_Filterator __begin, _Filterator __end, const _Tp &__old_value, const _Tp &__new_value)`
- `template<typename _Filter, typename _Predicate, typename _Tp >`
`void replace_if (_Filter, _Filter, _Predicate, const _Tp &)`
- `template<typename _Filter, typename _Predicate, typename _Tp >`
`void replace_if (_Filter, _Filter, _Predicate, const _Tp &, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Predicate, typename _Tp >`
`void replace_if (_Filter, _Filter, _Predicate, const _Tp &, __gnu_parallel::Parallelism)`
- `template<typename _Filterator, typename _Predicate, typename _Tp >`
`void replace_if (_Filterator __begin, _Filterator __end, _Predicate __pred, const _Tp &__new_value, __gnu_parallel::sequential_tag)`
- `template<typename _Filterator, typename _Predicate, typename _Tp >`
`void replace_if (_Filterator __begin, _Filterator __end, _Predicate __pred, const _Tp &__new_value, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filterator, typename _Predicate, typename _Tp >`
`void replace_if (_Filterator __begin, _Filterator __end, _Predicate __pred, const _Tp &__new_value)`
- `template<typename _Filter1, typename _Filter2 >`
`_Filter1 search (_Filter1, _Filter1, _Filter2, _Filter2, __gnu_parallel::sequential_tag)`

- `template<typename _Filter1, typename _Filter2 >`
`_Filter1 search (_Filter1, _Filter1, _Filter2, _Filter2)`
- `template<typename _Filter1, typename _Filter2, typename _BiPredicate >`
`_Filter1 search (_Filter1, _Filter1, _Filter2, _Filter2, _BiPredicate, __gnu_parallel::sequential_tag)`
- `template<typename _Filter1, typename _Filter2, typename _BiPredicate >`
`_Filter1 search (_Filter1, _Filter1, _Filter2, _Filter2, _BiPredicate)`
- `template<typename _FIterator1, typename _FIterator2 >`
`_FIterator1 search (_FIterator1 __begin1, _FIterator1 __end1, _FIterator2 __begin2, _FIterator2 __end2, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator1, typename _FIterator2 >`
`_FIterator1 search (_FIterator1 __begin1, _FIterator1 __end1, _FIterator2 __begin2, _FIterator2 __end2)`
- `template<typename _FIterator1, typename _FIterator2, typename _BinaryPredicate >`
`_FIterator1 search (_FIterator1 __begin1, _FIterator1 __end1, _FIterator2 __begin2, _FIterator2 __end2, __gnu_parallel::sequential_tag, __gnu_parallel::sequential_tag, _BinaryPredicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator1, typename _FIterator2, typename _BinaryPredicate >`
`_FIterator1 search (_FIterator1 __begin1, _FIterator1 __end1, _FIterator2 __begin2, _FIterator2 __end2, __gnu_parallel::sequential_tag, _BinaryPredicate __pred)`
- `template<typename _Filter, typename _Integer, typename _Tp >`
`_Filter search_n (_Filter, _Filter, _Integer, const _Tp &, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Integer, typename _Tp, typename _BiPredicate >`
`_Filter search_n (_Filter, _Filter, _Integer, const _Tp &, _BiPredicate, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Integer, typename _Tp >`
`_Filter search_n (_Filter, _Filter, _Integer, const _Tp &)`
- `template<typename _Filter, typename _Integer, typename _Tp, typename _BiPredicate >`
`_Filter search_n (_Filter, _Filter, _Integer, const _Tp &, _BiPredicate)`
- `template<typename _FIterator, typename _Integer, typename _Tp >`
`_FIterator search_n (_FIterator __begin, _FIterator __end, _Integer __count, const _Tp & __val, __gnu_parallel::sequential_tag, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator, typename _Integer, typename _Tp, typename _BinaryPredicate >`
`_FIterator search_n (_FIterator __begin, _FIterator __end, _Integer __count, const _Tp & __val, _BinaryPredicate __binary_pred, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator, typename _Integer, typename _Tp >`
`_FIterator search_n (_FIterator __begin, _FIterator __end, _Integer __count, const _Tp & __val)`
- `template<typename _FIterator, typename _Integer, typename _Tp, typename _BinaryPredicate >`
`_FIterator search_n (_FIterator __begin, _FIterator __end, _Integer __count, const _Tp & __val, _BinaryPredicate __binary_pred)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator >`
`_OutputIterator set_difference (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 __end2, _OutputIterator __out, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator set_difference (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 __end2, _OutputIterator __out, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator >`
`_OutputIterator set_difference (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 __end2, _OutputIterator __out)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator set_difference (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 __end2, _OutputIterator __out, _Predicate __pred)`
- `template<typename _IIter1, typename _IIter2, typename _OIter >`
`_OIter set_difference (_IIter1, _IIter1, _IIter2, _IIter2, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _OIter, typename _Predicate >`
`_OIter set_difference (_IIter1, _IIter1, _IIter2, _IIter2, _OIter, _Predicate, __gnu_parallel::sequential_tag)`

- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter set_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter set_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator set_intersection (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator set_intersection (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator set_intersection (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator set_intersection (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, _Predicate __pred)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter set_intersection (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter set_intersection (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter set_intersection (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter set_intersection (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator set_symmetric_difference (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator set_symmetric_difference (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator set_symmetric_difference (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator set_symmetric_difference (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, _Predicate __pred)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter set_symmetric_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter set_symmetric_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter set_symmetric_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter set_symmetric_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator set_union (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator set_union (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, _Predicate __pred, __gnu_parallel::sequential_tag)`

- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator set_union (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator`
`__out)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator set_union (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator`
`__out, _Predicate __pred)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate)`
- `template<typename _RAIter >`
`void sort (_RAIter __begin, _RAIter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare >`
`void sort (_RAIter __begin, _RAIter __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare, typename _Parallelism >`
`void sort (_RAIter __begin, _RAIter __end, _Compare __comp, _Parallelism __parallelism)`
- `template<typename _RAIter >`
`void sort (_RAIter __begin, _RAIter __end)`
- `template<typename _RAIter >`
`void sort (_RAIter __begin, _RAIter __end, __gnu_parallel::default_parallel_tag __parallelism)`
- `template<typename _RAIter >`
`void sort (_RAIter __begin, _RAIter __end, __gnu_parallel::parallel_tag __parallelism)`
- `template<typename _RAIter >`
`void sort (_RAIter __begin, _RAIter __end, __gnu_parallel::multiway_mergesort_tag __parallelism)`
- `template<typename _RAIter >`
`void sort (_RAIter __begin, _RAIter __end, __gnu_parallel::multiway_mergesort_sampling_tag __parallelism)`
- `template<typename _RAIter >`
`void sort (_RAIter __begin, _RAIter __end, __gnu_parallel::multiway_mergesort_exact_tag __parallelism)`
- `template<typename _RAIter >`
`void sort (_RAIter __begin, _RAIter __end, __gnu_parallel::quicksort_tag __parallelism)`
- `template<typename _RAIter >`
`void sort (_RAIter __begin, _RAIter __end, __gnu_parallel::balanced_quicksort_tag __parallelism)`
- `template<typename _RAIter, typename _Compare >`
`void sort (_RAIter __begin, _RAIter __end, _Compare __comp)`
- `template<typename _RAIter >`
`void stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare >`
`void stable_sort (_RAIter __begin, _RAIter __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare, typename _Parallelism >`
`void stable_sort (_RAIter __begin, _RAIter __end, _Compare __comp, _Parallelism __parallelism)`
- `template<typename _RAIter >`
`void stable_sort (_RAIter __begin, _RAIter __end)`
- `template<typename _RAIter >`
`void stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::default_parallel_tag __parallelism)`
- `template<typename _RAIter >`
`void stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::parallel_tag __parallelism)`
- `template<typename _RAIter >`
`void stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::multiway_mergesort_tag __parallelism)`

- `template<typename _RAIter >`
`void stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::quicksort_tag __parallelism)`
- `template<typename _RAIter >`
`void stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::balanced_quicksort_tag __parallelism)`
- `template<typename _RAIter, typename _Compare >`
`void stable_sort (_RAIter __begin, _RAIter __end, _Compare __comp)`
- `template<typename _Iter, typename _OIter, typename _UnaryOperation >`
`_OIter transform (_Iter, _Iter, _OIter, _UnaryOperation)`
- `template<typename _Iter, typename _OIter, typename _UnaryOperation >`
`_OIter transform (_Iter, _Iter, _OIter, _UnaryOperation, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter, typename _UnaryOperation >`
`_OIter transform (_Iter, _Iter, _OIter, _UnaryOperation, __gnu_parallel::_Parallelism)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _BiOperation >`
`_OIter transform (_Iter1, _Iter1, _Iter2, _OIter, _BiOperation)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _BiOperation >`
`_OIter transform (_Iter1, _Iter1, _Iter2, _OIter, _BiOperation, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _BiOperation >`
`_OIter transform (_Iter1, _Iter1, _Iter2, _OIter, _BiOperation, __gnu_parallel::_Parallelism)`
- `template<typename _Iter, typename _OutputIterator, typename _UnaryOperation >`
`_OutputIterator transform (_Iter __begin, _Iter __end, _OutputIterator __result, _UnaryOperation __unary_op, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _UnaryOperation >`
`_OutputIterator transform (_Iter __begin, _Iter __end, _OutputIterator __result, _UnaryOperation __unary_op, __gnu_parallel::_Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _UnaryOperation >`
`_OutputIterator transform (_Iter __begin, _Iter __end, _OutputIterator __result, _UnaryOperation __unary_op)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator transform (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _OutputIterator __result, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator transform (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _OutputIterator __result, __gnu_parallel::_Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator transform (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _OutputIterator __result, __gnu_parallel::_Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator transform (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _OutputIterator __result, __gnu_parallel::_Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator unique_copy (_Iter __begin1, _Iter __end1, _OutputIterator __out, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _Predicate >`
`_OutputIterator unique_copy (_Iter __begin1, _Iter __end1, _OutputIterator __out, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator unique_copy (_Iter __begin1, _Iter __end1, _OutputIterator __out)`
- `template<typename _Iter, typename _OutputIterator, typename _Predicate >`
`_OutputIterator unique_copy (_Iter __begin1, _Iter __end1, _OutputIterator __out, _Predicate __pred)`
- `template<typename _Iter, typename _OIter >`
`_OIter unique_copy (_Iter, _Iter, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter, typename _Predicate >`
`_OIter unique_copy (_Iter, _Iter, _OIter, _Predicate, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter >`
`_OIter unique_copy (_Iter, _Iter, _OIter)`
- `template<typename _Iter, typename _OIter, typename _Predicate >`
`_OIter unique_copy (_Iter, _Iter, _OIter, _Predicate)`

4.14.1 Detailed Description

GNU parallel code, replaces standard behavior with parallel behavior.

4.15 std::__profile Namespace Reference

Classes

- class [bitset](#)
- class [deque](#)
- class [forward_list](#)
- class [list](#)
- class [map](#)
- class [multimap](#)
- class [multiset](#)
- class [set](#)
- class [unordered_map](#)
- class [unordered_multimap](#)
- class [unordered_multiset](#)
- class [unordered_set](#)

Functions

- template<typename _UnorderedCont, typename _Value, bool _Cache_hash_code>
bool **are_equal** (const _UnorderedCont &__uc, const [__detail::Hash_node](#)< _Value, _Cache_hash_code >
*__lhs, const [__detail::Hash_node](#)< _Value, _Cache_hash_code > *__rhs)
- template<typename _UnorderedCont, typename _Value, bool _Cache_hash_code>
std::size_t **get_bucket_index** (const _UnorderedCont &__uc, const [__detail::Hash_node](#)< _Value, _Cache_↵
_hash_code > *__node)
- template<typename _Tp, typename _Alloc >
bool **operator!=** (const [deque](#)< _Tp, _Alloc > &__lhs, const [deque](#)< _Tp, _Alloc > &__rhs)
- template<typename _Tp, typename _Alloc >
bool **operator!=** (const [forward_list](#)< _Tp, _Alloc > &__lx, const [forward_list](#)< _Tp, _Alloc > &__ly)
- template<typename _IteratorL, typename _IteratorR, typename _Sequence >
bool **operator!=** (const [__iterator_tracker](#)< _IteratorL, _Sequence > &__lhs, const [__iterator_tracker](#)< _↵
IteratorR, _Sequence > &__rhs) noexcept
- template<typename _Iterator, typename _Sequence >
bool **operator!=** (const [__iterator_tracker](#)< _Iterator, _Sequence > &__lhs, const [__iterator_tracker](#)< _Iterator,
_Sequence > &__rhs) noexcept
- template<typename _Key, typename _Hash, typename _Pred, typename _Alloc >
bool **operator!=** (const [unordered_set](#)< _Key, _Hash, _Pred, _Alloc > &__x, const [unordered_set](#)< _Key, _↵
Hash, _Pred, _Alloc > &__y)
- template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc >
bool **operator!=** (const [unordered_map](#)< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, const [unordered_map](#)<
_Key, _Tp, _Hash, _Pred, _Alloc > &__y)
- template<typename _Tp, typename _Alloc >
bool **operator!=** (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)
- template<typename _Value, typename _Hash, typename _Pred, typename _Alloc >
bool **operator!=** (const [unordered_multiset](#)< _Value, _Hash, _Pred, _Alloc > &__x, const [unordered_multiset](#)<
_Value, _Hash, _Pred, _Alloc > &__y)

- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc >`
`bool operator!= (const unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, const unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator!= (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator!= (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator!= (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator!= (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator!= (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<size_t _Nb>`
`bitset< _Nb > operator& (const bitset< _Nb > &__x, const bitset< _Nb > &__y) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`__iterator_tracker< _Iterator, _Sequence > operator+ (typename __iterator_tracker< _Iterator, _Sequence >::difference_type __n, const __iterator_tracker< _Iterator, _Sequence > &__i) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`__iterator_tracker< _IteratorL, _Sequence >::difference_type operator- (const __iterator_tracker< _IteratorL, _Sequence > &__lhs, const __iterator_tracker< _IteratorR, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`__iterator_tracker< _Iterator, _Sequence >::difference_type operator- (const __iterator_tracker< _Iterator, _Sequence > &__lhs, const __iterator_tracker< _Iterator, _Sequence > &__rhs) noexcept`
- `template<typename _Tp, typename _Alloc >`
`bool operator< (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator< (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`bool operator< (const __iterator_tracker< _IteratorL, _Sequence > &__lhs, const __iterator_tracker< _IteratorR, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`bool operator< (const __iterator_tracker< _Iterator, _Sequence > &__lhs, const __iterator_tracker< _Iterator, _Sequence > &__rhs) noexcept`
- `template<typename _Tp, typename _Alloc >`
`bool operator< (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator< (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator< (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator< (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator< (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator< (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare, _Allocator > &__rhs)`

- `template<typename _CharT, typename _Traits, size_t _Nb>
std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const
bitset< _Nb > &__x)`
- `template<typename _Tp, typename _Alloc >
bool operator<= (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool operator<= (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >
bool operator<= (const __iterator_tracker< _IteratorL, _Sequence > &__lhs, const __iterator_tracker< _IteratorR, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >
bool operator<= (const __iterator_tracker< _Iterator, _Sequence > &__lhs, const __iterator_tracker< _Iterator, _Sequence > &__rhs) noexcept`
- `template<typename _Tp, typename _Alloc >
bool operator<= (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >
bool operator<= (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >
bool operator<= (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool operator<= (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >
bool operator<= (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >
bool operator<= (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool operator== (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool operator== (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >
bool operator== (const __iterator_tracker< _IteratorL, _Sequence > &__lhs, const __iterator_tracker< _IteratorR, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >
bool operator== (const __iterator_tracker< _Iterator, _Sequence > &__lhs, const __iterator_tracker< _Iterator, _Sequence > &__rhs) noexcept`
- `template<typename _Key, typename _Hash, typename _Pred, typename _Alloc >
bool operator== (const unordered_set< _Key, _Hash, _Pred, _Alloc > &__x, const unordered_set< _Key, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc >
bool operator== (const unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, const unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >
bool operator== (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc >
bool operator== (const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc >
bool operator== (const unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, const unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__y)`

- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator== (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator >`
`&__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator== (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _`
`Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator== (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp,`
`_Compare, _Allocator > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator== (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator== (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare,`
`_Allocator > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator> (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator> (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`bool operator> (const __iterator_tracker< _IteratorL, _Sequence > &__lhs, const __iterator_tracker< _`
`IteratorR, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`bool operator> (const __iterator_tracker< _Iterator, _Sequence > &__lhs, const __iterator_tracker< _Iterator,`
`_Sequence > &__rhs) noexcept`
- `template<typename _Tp, typename _Alloc >`
`bool operator> (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator> (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator >`
`&__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator> (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _`
`Allocator > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator> (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator> (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp,`
`_Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator> (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare,`
`_Allocator > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator>= (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator>= (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`bool operator>= (const __iterator_tracker< _IteratorL, _Sequence > &__lhs, const __iterator_tracker< _`
`IteratorR, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`bool operator>= (const __iterator_tracker< _Iterator, _Sequence > &__lhs, const __iterator_tracker< _Iterator,`
`_Sequence > &__rhs) noexcept`
- `template<typename _Tp, typename _Alloc >`
`bool operator>= (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`

- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator>= (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator >`
`&__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool operator>= (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare,`
`_Allocator > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool operator>= (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator>= (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp,`
`_Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool operator>= (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare,`
`_Allocator > &__rhs)`
- `template<typename _CharT, typename _Traits, size_t _Nb>`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, bitset<`
`_Nb > &__x)`
- `template<size_t _Nb>`
`bitset< _Nb > operator^ (const bitset< _Nb > &__x, const bitset< _Nb > &__y) noexcept`
- `template<size_t _Nb>`
`bitset< _Nb > operator| (const bitset< _Nb > &__x, const bitset< _Nb > &__y) noexcept`
- `template<typename _Tp, typename _Alloc >`
`void swap (deque< _Tp, _Alloc > &__lhs, deque< _Tp, _Alloc > &__rhs) noexcept(/*conditional */)`
- `template<typename _Tp, typename _Alloc >`
`void swap (forward_list< _Tp, _Alloc > &__lx, forward_list< _Tp, _Alloc > &__ly) noexcept(noexcept(__lx.<-`
`swap(__ly)))`
- `template<typename _Key, typename _Hash, typename _Pred, typename _Alloc >`
`void swap (unordered_set< _Key, _Hash, _Pred, _Alloc > &__x, unordered_set< _Key, _Hash, _Pred, _Alloc >`
`&__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc >`
`void swap (unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, unordered_map< _Key, _Tp, _Hash,`
`_Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc >`
`void swap (unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, unordered_multiset< _Value, _Hash,`
`_Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Tp, typename _Alloc >`
`void swap (vector< _Tp, _Alloc > &__lhs, vector< _Tp, _Alloc > &__rhs) noexcept(/*conditional */)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc >`
`void swap (unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > &__x, unordered_multimap< _Key, _Tp,`
`_Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`void swap (set< _Key, _Compare, _Allocator > &__x, set< _Key, _Compare, _Allocator > &__y) noexcept(/*conditional */)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`void swap (multiset< _Key, _Compare, _Allocator > &__x, multiset< _Key, _Compare, _Allocator > &__y) noexcept(/*conditional */)`
- `template<typename _Tp, typename _Alloc >`
`void swap (list< _Tp, _Alloc > &__lhs, list< _Tp, _Alloc > &__rhs) noexcept(/*conditional */)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`void swap (multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, multimap< _Key, _Tp, _Compare, _Allocator`
`> &__rhs) noexcept(/*conditional */)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`void swap (map< _Key, _Tp, _Compare, _Allocator > &__lhs, map< _Key, _Tp, _Compare, _Allocator > &__rhs)`
`noexcept(/*conditional */)`

4.15.1 Detailed Description

GNU profile code, replaces standard behavior with profile behavior.

4.15.2 Function Documentation

4.15.2.1 `template<typename _Tp, typename _Alloc> bool std::__profile::operator<= (const forward_list< _Tp, _Alloc > & __lx, const forward_list< _Tp, _Alloc > & __ly) [inline]`

Based on operator<.

Definition at line 202 of file profile/forward_list.

4.15.2.2 `template<typename _Tp, typename _Alloc> bool std::__profile::operator> (const forward_list< _Tp, _Alloc > & __lx, const forward_list< _Tp, _Alloc > & __ly) [inline]`

Based on operator<.

Definition at line 188 of file profile/forward_list.

4.15.2.3 `template<typename _Tp, typename _Alloc> bool std::__profile::operator>= (const forward_list< _Tp, _Alloc > & __lx, const forward_list< _Tp, _Alloc > & __ly) [inline]`

Based on operator<.

Definition at line 195 of file profile/forward_list.

4.15.2.4 `template<typename _Tp, typename _Alloc> void std::__profile::swap (forward_list< _Tp, _Alloc > & __lx, forward_list< _Tp, _Alloc > & __ly) [inline], [noexcept]`

See std::forward_list::swap().

Definition at line 209 of file profile/forward_list.

4.16 std::chrono Namespace Reference

Classes

- struct [duration](#)
- struct [duration_values](#)
- struct [time_point](#)
- struct [treat_as_floating_point](#)

Typedefs

- typedef `duration`< int64_t, ratio< 3600 > > `hours`
- typedef `duration`< int64_t, micro > `microseconds`
- typedef `duration`< int64_t, milli > `milliseconds`
- typedef `duration`< int64_t, ratio< 60 > > `minutes`
- typedef `duration`< int64_t, nano > `nanoseconds`
- typedef `duration`< int64_t > `seconds`

Functions

- template<typename _ToDur , typename _Rep , typename _Period >
constexpr enable_if< __is_duration< _ToDur >::value, _ToDur >::type `duration_cast` (const `duration`< _Rep, _Period > &__d)
- template<typename _Rep1 , typename _Period1 , typename _Rep2 , typename _Period2 >
constexpr bool **operator!=** (const `duration`< _Rep1, _Period1 > &__lhs, const `duration`< _Rep2, _Period2 > &__rhs)
- template<typename _Clock , typename _Dur1 , typename _Dur2 >
constexpr bool **operator!=** (const `time_point`< _Clock, _Dur1 > &__lhs, const `time_point`< _Clock, _Dur2 > &__rhs)
- template<typename _Rep1 , typename _Period , typename _Rep2 >
constexpr `duration`< typename __common_rep_type< _Rep1, typename enable_if<!__is_duration< _Rep2 >::value, _Rep2 >::type >::type, _Period > **operator%** (const `duration`< _Rep1, _Period > &__d, const _Rep2 &__s)
- template<typename _Rep1 , typename _Period1 , typename _Rep2 , typename _Period2 >
constexpr common_type< `duration`< _Rep1, _Period1 >, `duration`< _Rep2, _Period2 > >::type **operator%** (const `duration`< _Rep1, _Period1 > &__lhs, const `duration`< _Rep2, _Period2 > &__rhs)
- template<typename _Rep1 , typename _Period , typename _Rep2 >
constexpr `duration`< typename __common_rep_type< _Rep1, _Rep2 >::type, _Period > **operator*** (const `duration`< _Rep1, _Period > &__d, const _Rep2 &__s)
- template<typename _Rep1 , typename _Rep2 , typename _Period >
constexpr `duration`< typename __common_rep_type< _Rep2, _Rep1 >::type, _Period > **operator*** (const _Rep1 &__s, const `duration`< _Rep2, _Period > &__d)
- template<typename _Rep1 , typename _Period1 , typename _Rep2 , typename _Period2 >
constexpr common_type< `duration`< _Rep1, _Period1 >, `duration`< _Rep2, _Period2 > >::type **operator+** (const `duration`< _Rep1, _Period1 > &__lhs, const `duration`< _Rep2, _Period2 > &__rhs)
- template<typename _Clock , typename _Dur1 , typename _Rep2 , typename _Period2 >
constexpr `time_point`< _Clock, typename common_type< _Dur1, `duration`< _Rep2, _Period2 > >::type > **operator+** (const `time_point`< _Clock, _Dur1 > &__lhs, const `duration`< _Rep2, _Period2 > &__rhs)
- template<typename _Rep1 , typename _Period1 , typename _Clock , typename _Dur2 >
constexpr `time_point`< _Clock, typename common_type< `duration`< _Rep1, _Period1 >, _Dur2 >::type > **operator+** (const `duration`< _Rep1, _Period1 > &__lhs, const `time_point`< _Clock, _Dur2 > &__rhs)
- template<typename _Rep1 , typename _Period1 , typename _Rep2 , typename _Period2 >
constexpr common_type< `duration`< _Rep1, _Period1 >, `duration`< _Rep2, _Period2 > >::type **operator-** (const `duration`< _Rep1, _Period1 > &__lhs, const `duration`< _Rep2, _Period2 > &__rhs)
- template<typename _Clock , typename _Dur1 , typename _Rep2 , typename _Period2 >
constexpr `time_point`< _Clock, typename common_type< _Dur1, `duration`< _Rep2, _Period2 > >::type > **operator-** (const `time_point`< _Clock, _Dur1 > &__lhs, const `duration`< _Rep2, _Period2 > &__rhs)
- template<typename _Clock , typename _Dur1 , typename _Dur2 >
constexpr common_type< _Dur1, _Dur2 >::type **operator-** (const `time_point`< _Clock, _Dur1 > &__lhs, const `time_point`< _Clock, _Dur2 > &__rhs)

- `template<typename _Rep1, typename _Period, typename _Rep2 >`
`constexpr duration< typename __common_rep_type< _Rep1, typename enable_if<!__is_duration< _Rep2 >::value, _Rep2 >::type >::type, _Period > operator/ (const duration< _Rep1, _Period > &__d, const _Rep2 &__s)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr common_type< _Rep1, _Rep2 >::type operator/ (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool operator< (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool operator< (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool operator<= (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool operator<= (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool operator== (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool operator== (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool operator> (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool operator> (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool operator>= (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool operator>= (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 > &__rhs)`
- `template<typename _ToDur, typename _Clock, typename _Dur >`
`constexpr enable_if< __is_duration< _ToDur >::value, time_point< _Clock, _ToDur > >::type time_point_cast (const time_point< _Clock, _Dur > &__t)`

4.16.1 Detailed Description

ISO C++ 2011 entities sub-namespace for time and date.

4.16.2 Typedef Documentation

4.16.2.1 typedef duration<int64_t, ratio<3600> > std::chrono::hours

hours

Definition at line 542 of file chrono.

4.16.2.2 `typedef duration<int64_t, micro> std::chrono::microseconds`

microseconds

Definition at line 530 of file chrono.

4.16.2.3 `typedef duration<int64_t, milli> std::chrono::milliseconds`

milliseconds

Definition at line 533 of file chrono.

4.16.2.4 `typedef duration<int64_t, ratio< 60> > std::chrono::minutes`

minutes

Definition at line 539 of file chrono.

4.16.2.5 `typedef duration<int64_t, nano> std::chrono::nanoseconds`

nanoseconds

Definition at line 527 of file chrono.

4.16.2.6 `typedef duration<int64_t> std::chrono::seconds`

seconds

Definition at line 536 of file chrono.

4.16.3 Function Documentation

4.16.3.1 `template<typename _ToDur, typename _Rep, typename _Period> constexpr enable_if<__is_duration<_ToDur>::value, _ToDur>::type std::chrono::duration_cast(const duration<_Rep, _Period> & __d)`

duration_cast

Definition at line 194 of file chrono.

4.16.3.2 `template<typename _ToDur, typename _Clock, typename _Dur> constexpr enable_if<__is_duration<_ToDur>::value, time_point<_Clock, _ToDur>>::type std::chrono::time_point_cast(const time_point<_Clock, _Dur> & __t)`

time_point_cast

Definition at line 603 of file chrono.

4.17 std::decimal Namespace Reference

Classes

- class [decimal128](#)
- class [decimal32](#)
- class [decimal64](#)

Functions

- double **decimal128_to_double** ([decimal128](#) __d)
- float **decimal128_to_float** ([decimal128](#) __d)
- long double **decimal128_to_long_double** ([decimal128](#) __d)
- long long **decimal128_to_long_long** ([decimal128](#) __d)
- double **decimal32_to_double** ([decimal32](#) __d)
- float **decimal32_to_float** ([decimal32](#) __d)
- long double **decimal32_to_long_double** ([decimal32](#) __d)
- long long **decimal32_to_long_long** ([decimal32](#) __d)
- double **decimal64_to_double** ([decimal64](#) __d)
- float **decimal64_to_float** ([decimal64](#) __d)
- long double **decimal64_to_long_double** ([decimal64](#) __d)
- long long **decimal64_to_long_long** ([decimal64](#) __d)
- double **decimal_to_double** ([decimal32](#) __d)
- double **decimal_to_double** ([decimal64](#) __d)
- double **decimal_to_double** ([decimal128](#) __d)
- float **decimal_to_float** ([decimal32](#) __d)
- float **decimal_to_float** ([decimal64](#) __d)
- float **decimal_to_float** ([decimal128](#) __d)
- long double **decimal_to_long_double** ([decimal32](#) __d)
- long double **decimal_to_long_double** ([decimal64](#) __d)
- long double **decimal_to_long_double** ([decimal128](#) __d)
- long long **decimal_to_long_long** ([decimal32](#) __d)
- long long **decimal_to_long_long** ([decimal64](#) __d)
- long long **decimal_to_long_long** ([decimal128](#) __d)
- static [decimal128](#) **make_decimal128** (long long __coeff, int __exp)
- static [decimal128](#) **make_decimal128** (unsigned long long __coeff, int __exp)
- static [decimal32](#) **make_decimal32** (long long __coeff, int __exp)
- static [decimal32](#) **make_decimal32** (unsigned long long __coeff, int __exp)
- static [decimal64](#) **make_decimal64** (long long __coeff, int __exp)
- static [decimal64](#) **make_decimal64** (unsigned long long __coeff, int __exp)
- bool **operator!=** ([decimal32](#) __lhs, [decimal32](#) __rhs)
- bool **operator!=** ([decimal32](#) __lhs, [decimal64](#) __rhs)
- bool **operator!=** ([decimal32](#) __lhs, [decimal128](#) __rhs)
- bool **operator!=** ([decimal32](#) __lhs, int __rhs)
- bool **operator!=** ([decimal32](#) __lhs, unsigned int __rhs)
- bool **operator!=** ([decimal32](#) __lhs, long __rhs)
- bool **operator!=** ([decimal32](#) __lhs, unsigned long __rhs)
- bool **operator!=** ([decimal32](#) __lhs, long long __rhs)
- bool **operator!=** ([decimal32](#) __lhs, unsigned long long __rhs)
- bool **operator!=** (int __lhs, [decimal32](#) __rhs)

- bool **operator!=** (unsigned int __lhs, decimal32 __rhs)
- bool **operator!=** (long __lhs, decimal32 __rhs)
- bool **operator!=** (unsigned long __lhs, decimal32 __rhs)
- bool **operator!=** (long long __lhs, decimal32 __rhs)
- bool **operator!=** (unsigned long long __lhs, decimal32 __rhs)
- bool **operator!=** (decimal64 __lhs, decimal32 __rhs)
- bool **operator!=** (decimal64 __lhs, decimal64 __rhs)
- bool **operator!=** (decimal64 __lhs, decimal128 __rhs)
- bool **operator!=** (decimal64 __lhs, int __rhs)
- bool **operator!=** (decimal64 __lhs, unsigned int __rhs)
- bool **operator!=** (decimal64 __lhs, long __rhs)
- bool **operator!=** (decimal64 __lhs, unsigned long __rhs)
- bool **operator!=** (decimal64 __lhs, long long __rhs)
- bool **operator!=** (decimal64 __lhs, unsigned long long __rhs)
- bool **operator!=** (int __lhs, decimal64 __rhs)
- bool **operator!=** (unsigned int __lhs, decimal64 __rhs)
- bool **operator!=** (long __lhs, decimal64 __rhs)
- bool **operator!=** (unsigned long __lhs, decimal64 __rhs)
- bool **operator!=** (long long __lhs, decimal64 __rhs)
- bool **operator!=** (unsigned long long __lhs, decimal64 __rhs)
- bool **operator!=** (decimal128 __lhs, decimal32 __rhs)
- bool **operator!=** (decimal128 __lhs, decimal64 __rhs)
- bool **operator!=** (decimal128 __lhs, decimal128 __rhs)
- bool **operator!=** (decimal128 __lhs, int __rhs)
- bool **operator!=** (decimal128 __lhs, unsigned int __rhs)
- bool **operator!=** (decimal128 __lhs, long __rhs)
- bool **operator!=** (decimal128 __lhs, unsigned long __rhs)
- bool **operator!=** (decimal128 __lhs, long long __rhs)
- bool **operator!=** (decimal128 __lhs, unsigned long long __rhs)
- bool **operator!=** (int __lhs, decimal128 __rhs)
- bool **operator!=** (unsigned int __lhs, decimal128 __rhs)
- bool **operator!=** (long __lhs, decimal128 __rhs)
- bool **operator!=** (unsigned long __lhs, decimal128 __rhs)
- bool **operator!=** (long long __lhs, decimal128 __rhs)
- bool **operator!=** (unsigned long long __lhs, decimal128 __rhs)
- decimal32 **operator*** (decimal32 __lhs, decimal32 __rhs)
- decimal32 **operator*** (decimal32 __lhs, unsigned int __rhs)
- decimal32 **operator*** (decimal32 __lhs, int __rhs)
- decimal32 **operator*** (decimal32 __lhs, unsigned long __rhs)
- decimal32 **operator*** (decimal32 __lhs, long __rhs)
- decimal32 **operator*** (decimal32 __lhs, long long __rhs)
- decimal32 **operator*** (decimal32 __lhs, unsigned long long __rhs)
- decimal32 **operator*** (int __lhs, decimal32 __rhs)
- decimal32 **operator*** (unsigned int __lhs, decimal32 __rhs)
- decimal32 **operator*** (long __lhs, decimal32 __rhs)
- decimal32 **operator*** (unsigned long __lhs, decimal32 __rhs)
- decimal32 **operator*** (long long __lhs, decimal32 __rhs)
- decimal32 **operator*** (unsigned long long __lhs, decimal32 __rhs)
- decimal64 **operator*** (decimal32 __lhs, decimal64 __rhs)
- decimal64 **operator*** (decimal64 __lhs, decimal32 __rhs)
- decimal64 **operator*** (decimal64 __lhs, decimal64 __rhs)

- [decimal64 operator*](#) ([decimal64](#) __lhs, int __rhs)
- [decimal64 operator*](#) ([decimal64](#) __lhs, unsigned int __rhs)
- [decimal64 operator*](#) ([decimal64](#) __lhs, long __rhs)
- [decimal64 operator*](#) ([decimal64](#) __lhs, unsigned long __rhs)
- [decimal64 operator*](#) ([decimal64](#) __lhs, long long __rhs)
- [decimal64 operator*](#) ([decimal64](#) __lhs, unsigned long long __rhs)
- [decimal64 operator*](#) (int __lhs, [decimal64](#) __rhs)
- [decimal64 operator*](#) (unsigned int __lhs, [decimal64](#) __rhs)
- [decimal64 operator*](#) (long __lhs, [decimal64](#) __rhs)
- [decimal64 operator*](#) (unsigned long __lhs, [decimal64](#) __rhs)
- [decimal64 operator*](#) (long long __lhs, [decimal64](#) __rhs)
- [decimal64 operator*](#) (unsigned long long __lhs, [decimal64](#) __rhs)
- [decimal128 operator*](#) ([decimal32](#) __lhs, [decimal128](#) __rhs)
- [decimal128 operator*](#) ([decimal64](#) __lhs, [decimal128](#) __rhs)
- [decimal128 operator*](#) ([decimal128](#) __lhs, [decimal32](#) __rhs)
- [decimal128 operator*](#) ([decimal128](#) __lhs, [decimal64](#) __rhs)
- [decimal128 operator*](#) ([decimal128](#) __lhs, [decimal128](#) __rhs)
- [decimal128 operator*](#) ([decimal128](#) __lhs, int __rhs)
- [decimal128 operator*](#) ([decimal128](#) __lhs, unsigned int __rhs)
- [decimal128 operator*](#) ([decimal128](#) __lhs, long __rhs)
- [decimal128 operator*](#) ([decimal128](#) __lhs, unsigned long __rhs)
- [decimal128 operator*](#) ([decimal128](#) __lhs, long long __rhs)
- [decimal128 operator*](#) ([decimal128](#) __lhs, unsigned long long __rhs)
- [decimal128 operator*](#) (int __lhs, [decimal128](#) __rhs)
- [decimal128 operator*](#) (unsigned int __lhs, [decimal128](#) __rhs)
- [decimal128 operator*](#) (long __lhs, [decimal128](#) __rhs)
- [decimal128 operator*](#) (unsigned long __lhs, [decimal128](#) __rhs)
- [decimal128 operator*](#) (long long __lhs, [decimal128](#) __rhs)
- [decimal128 operator*](#) (unsigned long long __lhs, [decimal128](#) __rhs)
- [decimal32 operator+](#) ([decimal32](#) __rhs)
- [decimal64 operator+](#) ([decimal64](#) __rhs)
- [decimal128 operator+](#) ([decimal128](#) __rhs)
- [decimal32 operator+](#) ([decimal32](#) __lhs, [decimal32](#) __rhs)
- [decimal32 operator+](#) ([decimal32](#) __lhs, int __rhs)
- [decimal32 operator+](#) ([decimal32](#) __lhs, unsigned int __rhs)
- [decimal32 operator+](#) ([decimal32](#) __lhs, long __rhs)
- [decimal32 operator+](#) ([decimal32](#) __lhs, unsigned long __rhs)
- [decimal32 operator+](#) ([decimal32](#) __lhs, long long __rhs)
- [decimal32 operator+](#) ([decimal32](#) __lhs, unsigned long long __rhs)
- [decimal32 operator+](#) (int __lhs, [decimal32](#) __rhs)
- [decimal32 operator+](#) (unsigned int __lhs, [decimal32](#) __rhs)
- [decimal32 operator+](#) (long __lhs, [decimal32](#) __rhs)
- [decimal32 operator+](#) (unsigned long __lhs, [decimal32](#) __rhs)
- [decimal32 operator+](#) (long long __lhs, [decimal32](#) __rhs)
- [decimal32 operator+](#) (unsigned long long __lhs, [decimal32](#) __rhs)
- [decimal64 operator+](#) ([decimal32](#) __lhs, [decimal64](#) __rhs)
- [decimal64 operator+](#) ([decimal64](#) __lhs, [decimal32](#) __rhs)
- [decimal64 operator+](#) ([decimal64](#) __lhs, [decimal64](#) __rhs)
- [decimal64 operator+](#) (unsigned long long __lhs, [decimal64](#) __rhs)
- [decimal64 operator+](#) ([decimal64](#) __lhs, int __rhs)
- [decimal64 operator+](#) ([decimal64](#) __lhs, unsigned int __rhs)

- **decimal64 operator+** ([decimal64](#) __lhs, long __rhs)
- **decimal64 operator+** ([decimal64](#) __lhs, unsigned long __rhs)
- **decimal64 operator+** ([decimal64](#) __lhs, long long __rhs)
- **decimal64 operator+** ([decimal64](#) __lhs, unsigned long long __rhs)
- **decimal64 operator+** (int __lhs, [decimal64](#) __rhs)
- **decimal64 operator+** (unsigned int __lhs, [decimal64](#) __rhs)
- **decimal64 operator+** (long __lhs, [decimal64](#) __rhs)
- **decimal64 operator+** (unsigned long __lhs, [decimal64](#) __rhs)
- **decimal64 operator+** (long long __lhs, [decimal64](#) __rhs)
- **decimal128 operator+** ([decimal32](#) __lhs, [decimal128](#) __rhs)
- **decimal128 operator+** ([decimal64](#) __lhs, [decimal128](#) __rhs)
- **decimal128 operator+** ([decimal128](#) __lhs, [decimal32](#) __rhs)
- **decimal128 operator+** ([decimal128](#) __lhs, [decimal64](#) __rhs)
- **decimal128 operator+** ([decimal128](#) __lhs, [decimal128](#) __rhs)
- **decimal128 operator+** ([decimal128](#) __lhs, int __rhs)
- **decimal128 operator+** ([decimal128](#) __lhs, unsigned int __rhs)
- **decimal128 operator+** ([decimal128](#) __lhs, long __rhs)
- **decimal128 operator+** ([decimal128](#) __lhs, unsigned long __rhs)
- **decimal128 operator+** ([decimal128](#) __lhs, long long __rhs)
- **decimal128 operator+** ([decimal128](#) __lhs, unsigned long long __rhs)
- **decimal128 operator+** (int __lhs, [decimal128](#) __rhs)
- **decimal128 operator+** (unsigned int __lhs, [decimal128](#) __rhs)
- **decimal128 operator+** (long __lhs, [decimal128](#) __rhs)
- **decimal128 operator+** (unsigned long __lhs, [decimal128](#) __rhs)
- **decimal128 operator+** (long long __lhs, [decimal128](#) __rhs)
- **decimal128 operator+** (unsigned long long __lhs, [decimal128](#) __rhs)
- **decimal32 operator-** ([decimal32](#) __rhs)
- **decimal64 operator-** ([decimal64](#) __rhs)
- **decimal128 operator-** ([decimal128](#) __rhs)
- **decimal32 operator-** ([decimal32](#) __lhs, [decimal32](#) __rhs)
- **decimal32 operator-** ([decimal32](#) __lhs, int __rhs)
- **decimal32 operator-** ([decimal32](#) __lhs, unsigned int __rhs)
- **decimal32 operator-** ([decimal32](#) __lhs, long __rhs)
- **decimal32 operator-** ([decimal32](#) __lhs, unsigned long __rhs)
- **decimal32 operator-** ([decimal32](#) __lhs, long long __rhs)
- **decimal32 operator-** ([decimal32](#) __lhs, unsigned long long __rhs)
- **decimal32 operator-** (int __lhs, [decimal32](#) __rhs)
- **decimal32 operator-** (unsigned int __lhs, [decimal32](#) __rhs)
- **decimal32 operator-** (long __lhs, [decimal32](#) __rhs)
- **decimal32 operator-** (unsigned long __lhs, [decimal32](#) __rhs)
- **decimal32 operator-** (long long __lhs, [decimal32](#) __rhs)
- **decimal32 operator-** (unsigned long long __lhs, [decimal32](#) __rhs)
- **decimal64 operator-** ([decimal32](#) __lhs, [decimal64](#) __rhs)
- **decimal64 operator-** ([decimal64](#) __lhs, [decimal32](#) __rhs)
- **decimal64 operator-** ([decimal64](#) __lhs, [decimal64](#) __rhs)
- **decimal64 operator-** ([decimal64](#) __lhs, int __rhs)
- **decimal64 operator-** ([decimal64](#) __lhs, unsigned int __rhs)
- **decimal64 operator-** ([decimal64](#) __lhs, long __rhs)
- **decimal64 operator-** ([decimal64](#) __lhs, unsigned long __rhs)
- **decimal64 operator-** ([decimal64](#) __lhs, long long __rhs)
- **decimal64 operator-** ([decimal64](#) __lhs, unsigned long long __rhs)

- `decimal64 operator-` (int __lhs, `decimal64` __rhs)
- `decimal64 operator-` (unsigned int __lhs, `decimal64` __rhs)
- `decimal64 operator-` (long __lhs, `decimal64` __rhs)
- `decimal64 operator-` (unsigned long __lhs, `decimal64` __rhs)
- `decimal64 operator-` (long long __lhs, `decimal64` __rhs)
- `decimal64 operator-` (unsigned long long __lhs, `decimal64` __rhs)
- `decimal128 operator-` (`decimal32` __lhs, `decimal128` __rhs)
- `decimal128 operator-` (`decimal64` __lhs, `decimal128` __rhs)
- `decimal128 operator-` (`decimal128` __lhs, `decimal32` __rhs)
- `decimal128 operator-` (`decimal128` __lhs, `decimal64` __rhs)
- `decimal128 operator-` (`decimal128` __lhs, `decimal128` __rhs)
- `decimal128 operator-` (`decimal128` __lhs, int __rhs)
- `decimal128 operator-` (`decimal128` __lhs, unsigned int __rhs)
- `decimal128 operator-` (`decimal128` __lhs, long __rhs)
- `decimal128 operator-` (`decimal128` __lhs, unsigned long __rhs)
- `decimal128 operator-` (`decimal128` __lhs, long long __rhs)
- `decimal128 operator-` (`decimal128` __lhs, unsigned long long __rhs)
- `decimal128 operator-` (int __lhs, `decimal128` __rhs)
- `decimal128 operator-` (unsigned int __lhs, `decimal128` __rhs)
- `decimal128 operator-` (long __lhs, `decimal128` __rhs)
- `decimal128 operator-` (unsigned long __lhs, `decimal128` __rhs)
- `decimal128 operator-` (long long __lhs, `decimal128` __rhs)
- `decimal128 operator-` (unsigned long long __lhs, `decimal128` __rhs)
- `decimal32 operator/` (`decimal32` __lhs, `decimal32` __rhs)
- `decimal32 operator/` (`decimal32` __lhs, int __rhs)
- `decimal32 operator/` (`decimal32` __lhs, unsigned int __rhs)
- `decimal32 operator/` (`decimal32` __lhs, long __rhs)
- `decimal32 operator/` (`decimal32` __lhs, unsigned long __rhs)
- `decimal32 operator/` (`decimal32` __lhs, long long __rhs)
- `decimal32 operator/` (`decimal32` __lhs, unsigned long long __rhs)
- `decimal32 operator/` (int __lhs, `decimal32` __rhs)
- `decimal32 operator/` (unsigned int __lhs, `decimal32` __rhs)
- `decimal32 operator/` (long __lhs, `decimal32` __rhs)
- `decimal32 operator/` (unsigned long __lhs, `decimal32` __rhs)
- `decimal32 operator/` (long long __lhs, `decimal32` __rhs)
- `decimal32 operator/` (unsigned long long __lhs, `decimal32` __rhs)
- `decimal64 operator/` (`decimal32` __lhs, `decimal64` __rhs)
- `decimal64 operator/` (`decimal64` __lhs, `decimal32` __rhs)
- `decimal64 operator/` (`decimal64` __lhs, `decimal64` __rhs)
- `decimal64 operator/` (`decimal64` __lhs, int __rhs)
- `decimal64 operator/` (`decimal64` __lhs, unsigned int __rhs)
- `decimal64 operator/` (`decimal64` __lhs, long __rhs)
- `decimal64 operator/` (`decimal64` __lhs, unsigned long __rhs)
- `decimal64 operator/` (`decimal64` __lhs, long long __rhs)
- `decimal64 operator/` (`decimal64` __lhs, unsigned long long __rhs)
- `decimal64 operator/` (int __lhs, `decimal64` __rhs)
- `decimal64 operator/` (unsigned int __lhs, `decimal64` __rhs)
- `decimal64 operator/` (long __lhs, `decimal64` __rhs)
- `decimal64 operator/` (unsigned long __lhs, `decimal64` __rhs)
- `decimal64 operator/` (long long __lhs, `decimal64` __rhs)
- `decimal64 operator/` (unsigned long long __lhs, `decimal64` __rhs)

- **decimal128 operator/** ([decimal32](#) __lhs, [decimal128](#) __rhs)
- **decimal128 operator/** ([decimal64](#) __lhs, [decimal128](#) __rhs)
- **decimal128 operator/** ([decimal128](#) __lhs, [decimal32](#) __rhs)
- **decimal128 operator/** ([decimal128](#) __lhs, [decimal64](#) __rhs)
- **decimal128 operator/** ([decimal128](#) __lhs, [decimal128](#) __rhs)
- **decimal128 operator/** ([decimal128](#) __lhs, long __rhs)
- **decimal128 operator/** (long long __lhs, [decimal128](#) __rhs)
- **decimal128 operator/** ([decimal128](#) __lhs, int __rhs)
- **decimal128 operator/** ([decimal128](#) __lhs, unsigned int __rhs)
- **decimal128 operator/** ([decimal128](#) __lhs, unsigned long __rhs)
- **decimal128 operator/** ([decimal128](#) __lhs, long long __rhs)
- **decimal128 operator/** ([decimal128](#) __lhs, unsigned long long __rhs)
- **decimal128 operator/** (int __lhs, [decimal128](#) __rhs)
- **decimal128 operator/** (unsigned int __lhs, [decimal128](#) __rhs)
- **decimal128 operator/** (long __lhs, [decimal128](#) __rhs)
- **decimal128 operator/** (unsigned long __lhs, [decimal128](#) __rhs)
- **decimal128 operator/** (unsigned long long __lhs, [decimal128](#) __rhs)
- **bool operator<** (unsigned long __lhs, [decimal32](#) __rhs)
- **bool operator<** ([decimal32](#) __lhs, [decimal32](#) __rhs)
- **bool operator<** ([decimal32](#) __lhs, [decimal64](#) __rhs)
- **bool operator<** ([decimal32](#) __lhs, [decimal128](#) __rhs)
- **bool operator<** ([decimal32](#) __lhs, int __rhs)
- **bool operator<** ([decimal32](#) __lhs, long __rhs)
- **bool operator<** ([decimal32](#) __lhs, unsigned long __rhs)
- **bool operator<** ([decimal32](#) __lhs, long long __rhs)
- **bool operator<** (int __lhs, [decimal32](#) __rhs)
- **bool operator<** (long __lhs, [decimal32](#) __rhs)
- **bool operator<** ([decimal32](#) __lhs, unsigned long long __rhs)
- **bool operator<** (long long __lhs, [decimal32](#) __rhs)
- **bool operator<** (unsigned long long __lhs, [decimal32](#) __rhs)
- **bool operator<** (unsigned int __lhs, [decimal32](#) __rhs)
- **bool operator<** ([decimal32](#) __lhs, unsigned int __rhs)
- **bool operator<** (long __lhs, [decimal64](#) __rhs)
- **bool operator<** (unsigned long __lhs, [decimal64](#) __rhs)
- **bool operator<** ([decimal64](#) __lhs, [decimal64](#) __rhs)
- **bool operator<** (unsigned long long __lhs, [decimal64](#) __rhs)
- **bool operator<** (long long __lhs, [decimal64](#) __rhs)
- **bool operator<** ([decimal64](#) __lhs, [decimal32](#) __rhs)
- **bool operator<** ([decimal64](#) __lhs, [decimal128](#) __rhs)
- **bool operator<** ([decimal64](#) __lhs, unsigned int __rhs)
- **bool operator<** ([decimal64](#) __lhs, int __rhs)
- **bool operator<** (int __lhs, [decimal64](#) __rhs)
- **bool operator<** (unsigned int __lhs, [decimal64](#) __rhs)
- **bool operator<** ([decimal64](#) __lhs, long long __rhs)
- **bool operator<** ([decimal64](#) __lhs, long __rhs)
- **bool operator<** ([decimal64](#) __lhs, unsigned long __rhs)
- **bool operator<** ([decimal64](#) __lhs, unsigned long long __rhs)
- **bool operator<** (unsigned long __lhs, [decimal128](#) __rhs)
- **bool operator<** ([decimal128](#) __lhs, unsigned long long __rhs)
- **bool operator<** ([decimal128](#) __lhs, unsigned int __rhs)
- **bool operator<** (unsigned long long __lhs, [decimal128](#) __rhs)

- bool **operator**< (decimal128 __lhs, decimal32 __rhs)
- bool **operator**< (int __lhs, decimal128 __rhs)
- bool **operator**< (unsigned int __lhs, decimal128 __rhs)
- bool **operator**< (long long __lhs, decimal128 __rhs)
- bool **operator**< (long __lhs, decimal128 __rhs)
- bool **operator**< (decimal128 __lhs, unsigned long __rhs)
- bool **operator**< (decimal128 __lhs, int __rhs)
- bool **operator**< (decimal128 __lhs, decimal64 __rhs)
- bool **operator**< (decimal128 __lhs, long __rhs)
- bool **operator**< (decimal128 __lhs, decimal128 __rhs)
- bool **operator**< (decimal128 __lhs, long long __rhs)
- bool **operator**== (decimal32 __lhs, unsigned long __rhs)
- bool **operator**== (decimal32 __lhs, decimal128 __rhs)
- bool **operator**== (decimal32 __lhs, decimal32 __rhs)
- bool **operator**== (decimal32 __lhs, decimal64 __rhs)
- bool **operator**== (decimal32 __lhs, int __rhs)
- bool **operator**== (decimal32 __lhs, unsigned int __rhs)
- bool **operator**== (decimal32 __lhs, long __rhs)
- bool **operator**== (decimal32 __lhs, long long __rhs)
- bool **operator**== (decimal32 __lhs, unsigned long long __rhs)
- bool **operator**== (int __lhs, decimal32 __rhs)
- bool **operator**== (unsigned int __lhs, decimal32 __rhs)
- bool **operator**== (long __lhs, decimal32 __rhs)
- bool **operator**== (unsigned long __lhs, decimal32 __rhs)
- bool **operator**== (long long __lhs, decimal32 __rhs)
- bool **operator**== (unsigned long long __lhs, decimal32 __rhs)
- bool **operator**== (unsigned long long __lhs, decimal64 __rhs)
- bool **operator**== (long __lhs, decimal64 __rhs)
- bool **operator**== (decimal64 __lhs, long long __rhs)
- bool **operator**== (decimal64 __lhs, unsigned int __rhs)
- bool **operator**== (decimal64 __lhs, decimal128 __rhs)
- bool **operator**== (long long __lhs, decimal64 __rhs)
- bool **operator**== (decimal64 __lhs, int __rhs)
- bool **operator**== (decimal64 __lhs, long __rhs)
- bool **operator**== (decimal64 __lhs, decimal32 __rhs)
- bool **operator**== (decimal64 __lhs, decimal64 __rhs)
- bool **operator**== (decimal64 __lhs, unsigned long __rhs)
- bool **operator**== (decimal64 __lhs, unsigned long long __rhs)
- bool **operator**== (int __lhs, decimal64 __rhs)
- bool **operator**== (unsigned int __lhs, decimal64 __rhs)
- bool **operator**== (unsigned long __lhs, decimal64 __rhs)
- bool **operator**== (int __lhs, decimal128 __rhs)
- bool **operator**== (unsigned int __lhs, decimal128 __rhs)
- bool **operator**== (long __lhs, decimal128 __rhs)
- bool **operator**== (long long __lhs, decimal128 __rhs)
- bool **operator**== (unsigned long long __lhs, decimal128 __rhs)
- bool **operator**== (unsigned long __lhs, decimal128 __rhs)
- bool **operator**== (decimal128 __lhs, decimal32 __rhs)
- bool **operator**== (decimal128 __lhs, unsigned int __rhs)
- bool **operator**== (decimal128 __lhs, unsigned long long __rhs)
- bool **operator**== (decimal128 __lhs, unsigned long __rhs)

- bool **operator==** (decimal128 __lhs, decimal128 __rhs)
- bool **operator==** (decimal128 __lhs, long long __rhs)
- bool **operator==** (decimal128 __lhs, decimal64 __rhs)
- bool **operator==** (decimal128 __lhs, int __rhs)
- bool **operator==** (decimal128 __lhs, long __rhs)
- bool **operator>** (unsigned int __lhs, decimal32 __rhs)
- bool **operator>** (long __lhs, decimal32 __rhs)
- bool **operator>** (decimal32 __lhs, decimal128 __rhs)
- bool **operator>** (decimal32 __lhs, long long __rhs)
- bool **operator>** (decimal32 __lhs, unsigned long long __rhs)
- bool **operator>** (decimal32 __lhs, unsigned long __rhs)
- bool **operator>** (decimal32 __lhs, decimal32 __rhs)
- bool **operator>** (decimal32 __lhs, decimal64 __rhs)
- bool **operator>** (decimal32 __lhs, long __rhs)
- bool **operator>** (unsigned long __lhs, decimal32 __rhs)
- bool **operator>** (unsigned long long __lhs, decimal32 __rhs)
- bool **operator>** (long long __lhs, decimal32 __rhs)
- bool **operator>** (decimal32 __lhs, unsigned int __rhs)
- bool **operator>** (int __lhs, decimal32 __rhs)
- bool **operator>** (decimal32 __lhs, int __rhs)
- bool **operator>** (decimal64 __lhs, unsigned long long __rhs)
- bool **operator>** (decimal64 __lhs, decimal32 __rhs)
- bool **operator>** (unsigned long __lhs, decimal64 __rhs)
- bool **operator>** (unsigned long long __lhs, decimal64 __rhs)
- bool **operator>** (long long __lhs, decimal64 __rhs)
- bool **operator>** (int __lhs, decimal64 __rhs)
- bool **operator>** (decimal64 __lhs, unsigned int __rhs)
- bool **operator>** (decimal64 __lhs, unsigned long __rhs)
- bool **operator>** (decimal64 __lhs, decimal128 __rhs)
- bool **operator>** (decimal64 __lhs, long __rhs)
- bool **operator>** (decimal64 __lhs, long long __rhs)
- bool **operator>** (decimal64 __lhs, decimal64 __rhs)
- bool **operator>** (decimal64 __lhs, int __rhs)
- bool **operator>** (long __lhs, decimal64 __rhs)
- bool **operator>** (unsigned int __lhs, decimal64 __rhs)
- bool **operator>** (decimal128 __lhs, decimal128 __rhs)
- bool **operator>** (int __lhs, decimal128 __rhs)
- bool **operator>** (decimal128 __lhs, unsigned long __rhs)
- bool **operator>** (unsigned long long __lhs, decimal128 __rhs)
- bool **operator>** (decimal128 __lhs, unsigned int __rhs)
- bool **operator>** (unsigned int __lhs, decimal128 __rhs)
- bool **operator>** (decimal128 __lhs, decimal32 __rhs)
- bool **operator>** (decimal128 __lhs, unsigned long long __rhs)
- bool **operator>** (decimal128 __lhs, long __rhs)
- bool **operator>** (unsigned long __lhs, decimal128 __rhs)
- bool **operator>** (decimal128 __lhs, int __rhs)
- bool **operator>** (long long __lhs, decimal128 __rhs)
- bool **operator>** (long __lhs, decimal128 __rhs)
- bool **operator>** (decimal128 __lhs, decimal64 __rhs)
- bool **operator>** (decimal128 __lhs, long long __rhs)
- bool **operator>=** (long long __lhs, decimal32 __rhs)

- bool **operator**>= (unsigned long __lhs, decimal32 __rhs)
- bool **operator**>= (decimal32 __lhs, decimal64 __rhs)
- bool **operator**>= (decimal32 __lhs, unsigned int __rhs)
- bool **operator**>= (decimal32 __lhs, decimal32 __rhs)
- bool **operator**>= (decimal32 __lhs, int __rhs)
- bool **operator**>= (decimal32 __lhs, decimal128 __rhs)
- bool **operator**>= (unsigned long long __lhs, decimal32 __rhs)
- bool **operator**>= (unsigned int __lhs, decimal32 __rhs)
- bool **operator**>= (long __lhs, decimal32 __rhs)
- bool **operator**>= (decimal32 __lhs, unsigned long long __rhs)
- bool **operator**>= (decimal32 __lhs, long long __rhs)
- bool **operator**>= (int __lhs, decimal32 __rhs)
- bool **operator**>= (decimal32 __lhs, long __rhs)
- bool **operator**>= (decimal32 __lhs, unsigned long __rhs)
- bool **operator**>= (unsigned long long __lhs, decimal64 __rhs)
- bool **operator**>= (decimal64 __lhs, unsigned long long __rhs)
- bool **operator**>= (decimal64 __lhs, long long __rhs)
- bool **operator**>= (decimal64 __lhs, decimal64 __rhs)
- bool **operator**>= (decimal64 __lhs, decimal32 __rhs)
- bool **operator**>= (decimal64 __lhs, unsigned int __rhs)
- bool **operator**>= (decimal64 __lhs, unsigned long __rhs)
- bool **operator**>= (decimal64 __lhs, decimal128 __rhs)
- bool **operator**>= (long __lhs, decimal64 __rhs)
- bool **operator**>= (decimal64 __lhs, long __rhs)
- bool **operator**>= (unsigned int __lhs, decimal64 __rhs)
- bool **operator**>= (decimal64 __lhs, int __rhs)
- bool **operator**>= (unsigned long __lhs, decimal64 __rhs)
- bool **operator**>= (int __lhs, decimal64 __rhs)
- bool **operator**>= (long long __lhs, decimal64 __rhs)
- bool **operator**>= (decimal128 __lhs, int __rhs)
- bool **operator**>= (int __lhs, decimal128 __rhs)
- bool **operator**>= (decimal128 __lhs, unsigned long __rhs)
- bool **operator**>= (long long __lhs, decimal128 __rhs)
- bool **operator**>= (decimal128 __lhs, decimal64 __rhs)
- bool **operator**>= (unsigned long __lhs, decimal128 __rhs)
- bool **operator**>= (decimal128 __lhs, decimal32 __rhs)
- bool **operator**>= (decimal128 __lhs, long __rhs)
- bool **operator**>= (decimal128 __lhs, unsigned int __rhs)
- bool **operator**>= (decimal128 __lhs, long long __rhs)
- bool **operator**>= (decimal128 __lhs, decimal128 __rhs)
- bool **operator**>= (unsigned int __lhs, decimal128 __rhs)
- bool **operator**>= (decimal128 __lhs, unsigned long long __rhs)
- bool **operator**>= (long __lhs, decimal128 __rhs)
- bool **operator**>= (unsigned long long __lhs, decimal128 __rhs)

4.17.1 Detailed Description

ISO/IEC TR 24733 Decimal floating-point arithmetic.

4.17.2 Function Documentation

4.17.2.1 `long long std::decimal::decimal32_to_long_long (decimal32 __d)`

Non-conforming extension: Conversion to integral type.

4.18 `std::placeholders` Namespace Reference

Variables

- `const _Placeholder< 1 > _1`
- `const _Placeholder< 10 > _10`
- `const _Placeholder< 11 > _11`
- `const _Placeholder< 12 > _12`
- `const _Placeholder< 13 > _13`
- `const _Placeholder< 14 > _14`
- `const _Placeholder< 15 > _15`
- `const _Placeholder< 16 > _16`
- `const _Placeholder< 17 > _17`
- `const _Placeholder< 18 > _18`
- `const _Placeholder< 19 > _19`
- `const _Placeholder< 2 > _2`
- `const _Placeholder< 20 > _20`
- `const _Placeholder< 21 > _21`
- `const _Placeholder< 22 > _22`
- `const _Placeholder< 23 > _23`
- `const _Placeholder< 24 > _24`
- `const _Placeholder< 25 > _25`
- `const _Placeholder< 26 > _26`
- `const _Placeholder< 27 > _27`
- `const _Placeholder< 28 > _28`
- `const _Placeholder< 29 > _29`
- `const _Placeholder< 3 > _3`
- `const _Placeholder< 4 > _4`
- `const _Placeholder< 5 > _5`
- `const _Placeholder< 6 > _6`
- `const _Placeholder< 7 > _7`
- `const _Placeholder< 8 > _8`
- `const _Placeholder< 9 > _9`

4.18.1 Detailed Description

ISO C++11 entities sub-namespace for functional.

4.19 std::regex_constants Namespace Reference

5.1 Regular Expression Syntax Options

- enum `__syntax_option` {
`_S_icode`, `_S_nosubs`, `_S_optimize`, `_S_collate`,
`_S_ECMAScript`, `_S_basic`, `_S_extended`, `_S_awk`,
`_S_grep`, `_S_egrep`, `_S_polynomial`, `_S_syntax_last` }
- enum `syntax_option_type` : unsigned int
- constexpr `syntax_option_type` `icase`
- constexpr `syntax_option_type` `nosubs`
- constexpr `syntax_option_type` `optimize`
- constexpr `syntax_option_type` `collate`
- constexpr `syntax_option_type` `ECMAScript`
- constexpr `syntax_option_type` `basic`
- constexpr `syntax_option_type` `extended`
- constexpr `syntax_option_type` `awk`
- constexpr `syntax_option_type` `grep`
- constexpr `syntax_option_type` `egrep`
- constexpr `syntax_option_type` `__polynomial`
- constexpr `syntax_option_type` `operator&` (`syntax_option_type` __a, `syntax_option_type` __b)
- constexpr `syntax_option_type` `operator|` (`syntax_option_type` __a, `syntax_option_type` __b)
- constexpr `syntax_option_type` `operator^` (`syntax_option_type` __a, `syntax_option_type` __b)
- constexpr `syntax_option_type` `operator~` (`syntax_option_type` __a)
- `syntax_option_type` & `operator&=` (`syntax_option_type` &__a, `syntax_option_type` __b)
- `syntax_option_type` & `operator|=` (`syntax_option_type` &__a, `syntax_option_type` __b)
- `syntax_option_type` & `operator^=` (`syntax_option_type` &__a, `syntax_option_type` __b)

5.2 Matching Rules

Matching a regular expression against a sequence of characters [first, last) proceeds according to the rules of the grammar specified for the regular expression object, modified according to the effects listed below for any bitmask elements set.

- enum `__match_flag` {
`_S_not_bol`, `_S_not_eol`, `_S_not_bow`, `_S_not_eow`,
`_S_any`, `_S_not_null`, `_S_continuous`, `_S_prev_avail`,
`_S_sed`, `_S_no_copy`, `_S_first_only`, `_S_match_flag_last` }
- enum `match_flag_type` : unsigned int
- constexpr `match_flag_type` `match_default`
- constexpr `match_flag_type` `match_not_bol`
- constexpr `match_flag_type` `match_not_eol`
- constexpr `match_flag_type` `match_not_bow`
- constexpr `match_flag_type` `match_not_eow`
- constexpr `match_flag_type` `match_any`
- constexpr `match_flag_type` `match_not_null`
- constexpr `match_flag_type` `match_continuous`
- constexpr `match_flag_type` `match_prev_avail`
- constexpr `match_flag_type` `format_default`

- constexpr [match_flag_type](#) format_sed
- constexpr [match_flag_type](#) format_no_copy
- constexpr [match_flag_type](#) format_first_only
- constexpr [match_flag_type](#) operator& ([match_flag_type](#) __a, [match_flag_type](#) __b)
- constexpr [match_flag_type](#) operator| ([match_flag_type](#) __a, [match_flag_type](#) __b)
- constexpr [match_flag_type](#) operator^ ([match_flag_type](#) __a, [match_flag_type](#) __b)
- constexpr [match_flag_type](#) operator~ ([match_flag_type](#) __a)
- [match_flag_type](#) & operator&= ([match_flag_type](#) &__a, [match_flag_type](#) __b)
- [match_flag_type](#) & operator|= ([match_flag_type](#) &__a, [match_flag_type](#) __b)
- [match_flag_type](#) & operator^= ([match_flag_type](#) &__a, [match_flag_type](#) __b)

5.3 Error Types

- enum [error_type](#) {
[_S_error_collate](#), [_S_error_ctype](#), [_S_error_escape](#), [_S_error_backref](#),
[_S_error_brack](#), [_S_error_paren](#), [_S_error_brace](#), [_S_error_badbrace](#),
[_S_error_range](#), [_S_error_space](#), [_S_error_badrepeat](#), [_S_error_complexity](#),
[_S_error_stack](#) }
- constexpr [error_type](#) [error_collate](#) ([_S_error_collate](#))
- constexpr [error_type](#) [error_ctype](#) ([_S_error_ctype](#))
- constexpr [error_type](#) [error_escape](#) ([_S_error_escape](#))
- constexpr [error_type](#) [error_backref](#) ([_S_error_backref](#))
- constexpr [error_type](#) [error_brack](#) ([_S_error_brack](#))
- constexpr [error_type](#) [error_paren](#) ([_S_error_paren](#))
- constexpr [error_type](#) [error_brace](#) ([_S_error_brace](#))
- constexpr [error_type](#) [error_badbrace](#) ([_S_error_badbrace](#))
- constexpr [error_type](#) [error_range](#) ([_S_error_range](#))
- constexpr [error_type](#) [error_space](#) ([_S_error_space](#))
- constexpr [error_type](#) [error_badrepeat](#) ([_S_error_badrepeat](#))
- constexpr [error_type](#) [error_complexity](#) ([_S_error_complexity](#))
- constexpr [error_type](#) [error_stack](#) ([_S_error_stack](#))

4.19.1 Detailed Description

ISO C++-0x entities sub namespace for regex.

4.19.2 Enumeration Type Documentation

4.19.2.1 enum std::regex_constants::__match_flag

This is a bitmask type indicating regex matching rules.

The [match_flag_type](#) is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

Definition at line 232 of file [regex_constants.h](#).

4.19.2.2 `enum std::regex_constants::__syntax_option`

This is a bitmask type indicating how to interpret the regex.

The `syntax_option_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

A valid value of type `syntax_option_type` shall have exactly one of the elements `ECMAScript`, `basic`, `extended`, `awk`, `grep`, `egrep` set.

Definition at line 54 of file `regex_constants.h`.

4.19.2.3 `enum std::regex_constants::error_type`

The expression contained an invalid collating element name.

Definition at line 49 of file `regex_error.h`.

4.19.2.4 `enum std::regex_constants::match_flag_type : unsigned int`

This is a bitmask type indicating regex matching rules.

The `match_flag_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

Definition at line 255 of file `regex_constants.h`.

4.19.2.5 `enum std::regex_constants::syntax_option_type : unsigned int`

This is a bitmask type indicating how to interpret the regex.

The `syntax_option_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

A valid value of type `syntax_option_type` shall have exactly one of the elements `ECMAScript`, `basic`, `extended`, `awk`, `grep`, `egrep` set.

Definition at line 81 of file `regex_constants.h`.

4.19.3 Function Documentation

4.19.3.1 `constexpr error_type std::regex_constants::error_backref (_S_error_backref)`

The expression contained an invalid back reference.

4.19.3.2 `constexpr error_type std::regex_constants::error_badbrace (_S_error_badbrace)`

The expression contained an invalid range in a `{}` expression.

4.19.3.3 `constexpr error_type std::regex_constants::error_badrepeat (_S_error_badrepeat)`

One of `*?+{` was not preceded by a valid regular expression.

4.19.3.4 `constexpr error_type std::regex_constants::error_brace (_S_error_brace)`

The expression contained mismatched `{` and `}`

4.19.3.5 `constexpr error_type std::regex_constants::error_brack (_S_error_brack)`

The expression contained mismatched `[` and `]`.

4.19.3.6 `constexpr error_type std::regex_constants::error_collate (_S_error_collate)`

The expression contained an invalid collating element name.

4.19.3.7 `constexpr error_type std::regex_constants::error_complexity (_S_error_complexity)`

The complexity of an attempted match against a regular expression exceeded a pre-set level.

4.19.3.8 `constexpr error_type std::regex_constants::error_ctype (_S_error_ctype)`

The expression contained an invalid character class name.

4.19.3.9 `constexpr error_type std::regex_constants::error_escape (_S_error_escape)`

The expression contained an invalid escaped character, or a trailing escape.

4.19.3.10 `constexpr error_type std::regex_constants::error_paren (_S_error_paren)`

The expression contained mismatched `(` and `)`.

4.19.3.11 `constexpr error_type std::regex_constants::error_range (_S_error_range)`

The expression contained an invalid character range, such as `[b-a]` in most encodings.

4.19.3.12 `constexpr error_type std::regex_constants::error_space (_S_error_space)`

There was insufficient memory to convert the expression into a finite state machine.

4.19.3.13 `constexpr error_type std::regex_constants::error_stack (_S_error_stack)`

There was insufficient memory to determine whether the regular expression could match the specified character sequence.

4.19.3.14 `constexpr syntax_option_type std::regex_constants::operator& (syntax_option_type __a,
syntax_option_type __b) [inline]`

This is a bitmask type indicating how to interpret the regex.

The `syntax_option_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

A valid value of type `syntax_option_type` shall have exactly one of the elements `ECMAScript`, `basic`, `extended`, `awk`, `grep`, `egrep` set.

Definition at line 183 of file `regex_constants.h`.

4.19.3.15 `constexpr match_flag_type std::regex_constants::operator& (match_flag_type __a, match_flag_type __b)
[inline]`

This is a bitmask type indicating regex matching rules.

The `match_flag_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

Definition at line 372 of file `regex_constants.h`.

4.19.3.16 `syntax_option_type& std::regex_constants::operator&= (syntax_option_type & __a, syntax_option_type __b
) [inline]`

This is a bitmask type indicating how to interpret the regex.

The `syntax_option_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

A valid value of type `syntax_option_type` shall have exactly one of the elements `ECMAScript`, `basic`, `extended`, `awk`, `grep`, `egrep` set.

Definition at line 208 of file `regex_constants.h`.

4.19.3.17 `match_flag_type& std::regex_constants::operator&= (match_flag_type & __a, match_flag_type __b)
[inline]`

This is a bitmask type indicating regex matching rules.

The `match_flag_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

Definition at line 397 of file `regex_constants.h`.

```
4.19.3.18 constexpr syntax_option_type std::regex_constants::operator^ ( syntax_option_type __a,
syntax_option_type __b ) [inline]
```

This is a bitmask type indicating how to interpret the regex.

The `syntax_option_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

A valid value of type `syntax_option_type` shall have exactly one of the elements `ECMAScript`, `basic`, `extended`, `awk`, `grep`, `egrep` set.

Definition at line 197 of file `regex_constants.h`.

```
4.19.3.19 constexpr match_flag_type std::regex_constants::operator^ ( match_flag_type __a, match_flag_type __b )
[inline]
```

This is a bitmask type indicating regex matching rules.

The `match_flag_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

Definition at line 386 of file `regex_constants.h`.

```
4.19.3.20 syntax_option_type& std::regex_constants::operator^= ( syntax_option_type & __a, syntax_option_type __b
) [inline]
```

This is a bitmask type indicating how to interpret the regex.

The `syntax_option_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

A valid value of type `syntax_option_type` shall have exactly one of the elements `ECMAScript`, `basic`, `extended`, `awk`, `grep`, `egrep` set.

Definition at line 216 of file `regex_constants.h`.

```
4.19.3.21 match_flag_type& std::regex_constants::operator^= ( match_flag_type & __a, match_flag_type __b )
[inline]
```

This is a bitmask type indicating regex matching rules.

The `match_flag_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

Definition at line 405 of file `regex_constants.h`.

4.19.3.22 `constexpr syntax_option_type std::regex_constants::operator|(syntax_option_type __a, syntax_option_type __b) [inline]`

This is a bitmask type indicating how to interpret the regex.

The `syntax_option_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

A valid value of type `syntax_option_type` shall have exactly one of the elements `ECMAScript`, `basic`, `extended`, `awk`, `grep`, `egrep` set.

Definition at line 190 of file `regex_constants.h`.

4.19.3.23 `constexpr match_flag_type std::regex_constants::operator|(match_flag_type __a, match_flag_type __b) [inline]`

This is a bitmask type indicating regex matching rules.

The `match_flag_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

Definition at line 379 of file `regex_constants.h`.

4.19.3.24 `syntax_option_type& std::regex_constants::operator|= (syntax_option_type & __a, syntax_option_type __b) [inline]`

This is a bitmask type indicating how to interpret the regex.

The `syntax_option_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

A valid value of type `syntax_option_type` shall have exactly one of the elements `ECMAScript`, `basic`, `extended`, `awk`, `grep`, `egrep` set.

Definition at line 212 of file `regex_constants.h`.

4.19.3.25 `match_flag_type& std::regex_constants::operator|= (match_flag_type & __a, match_flag_type __b) [inline]`

This is a bitmask type indicating regex matching rules.

The `match_flag_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

Definition at line 401 of file `regex_constants.h`.

4.19.3.26 `constexpr syntax_option_type std::regex_constants::operator~(syntax_option_type __a) [inline]`

This is a bitmask type indicating how to interpret the regex.

The `syntax_option_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

A valid value of type `syntax_option_type` shall have exactly one of the elements `ECMAScript`, `basic`, `extended`, `awk`, `grep`, `egrep` set.

Definition at line 204 of file `regex_constants.h`.

4.19.3.27 `constexpr match_flag_type std::regex_constants::operator~(match_flag_type __a) [inline]`

This is a bitmask type indicating regex matching rules.

The `match_flag_type` is implementation defined but it is valid to perform bitwise operations on these values and expect the right thing to happen.

Definition at line 393 of file `regex_constants.h`.

4.19.4 Variable Documentation

4.19.4.1 `constexpr syntax_option_type std::regex_constants::_polynomial`

Extension: Ensure both space complexity of compiled regex and time complexity execution are not exponential. If specified in a regex with back-references, the exception `regex_constants::error_complexity` will be thrown.

Definition at line 179 of file `regex_constants.h`.

4.19.4.2 `constexpr syntax_option_type std::regex_constants::awk`

Specifies that the grammar recognized by the regular expression engine is that used by POSIX utility `awk` in IEEE Std 1003.1-2001. This option is identical to `syntax_option_type extended`, except that C-style escape sequences are supported. These sequences are: `\\`, `\a`, `\b`, `\f`, `\n`, `\r`, `\t`, `\v`, `\'`, `\'`, and `\ddd` (where `ddd` is one, two, or three octal digits).

Definition at line 152 of file `regex_constants.h`.

Referenced by `std::regex_traits< _Ch_type >::getloc()`.

4.19.4.3 `constexpr syntax_option_type std::regex_constants::basic`

Specifies that the grammar recognized by the regular expression engine is that used by POSIX basic regular expressions in IEEE Std 1003.1-2001, Portable Operating System Interface (POSIX), Base Definitions and Headers, Section 9, Regular Expressions [IEEE, Information Technology – Portable Operating System Interface (POSIX), IEEE Standard 1003.1-2001].

Definition at line 132 of file `regex_constants.h`.

Referenced by `std::regex_traits< _Ch_type >::getloc()`.

4.19.4.4 `constexpr syntax_option_type std::regex_constants::collate`

Specifies that character ranges of the form `[a-b]` should be locale sensitive.

Definition at line 111 of file `regex_constants.h`.

Referenced by `std::regex_traits<_Ch_type>::getloc()`.

4.19.4.5 `constexpr syntax_option_type std::regex_constants::ECMAScript`

Specifies that the grammar recognized by the regular expression engine is that used by ECMAScript in ECMA-262 [Ecma International, ECMAScript Language Specification, Standard Ecma-262, third edition, 1999], as modified in section [28.13]. This grammar is similar to that defined in the PERL scripting language but extended with elements found in the POSIX regular expression grammar.

Definition at line 122 of file `regex_constants.h`.

Referenced by `std::regex_traits<_Ch_type>::getloc()`.

4.19.4.6 `constexpr syntax_option_type std::regex_constants::egrep`

Specifies that the grammar recognized by the regular expression engine is that used by POSIX utility `grep` when given the `-E` option in IEEE Std 1003.1-2001. This option is identical to `syntax_option_type` extended, except that newlines are treated as whitespace.

Definition at line 170 of file `regex_constants.h`.

Referenced by `std::regex_traits<_Ch_type>::getloc()`.

4.19.4.7 `constexpr syntax_option_type std::regex_constants::extended`

Specifies that the grammar recognized by the regular expression engine is that used by POSIX extended regular expressions in IEEE Std 1003.1-2001, Portable Operating System Interface (POSIX), Base Definitions and Headers, Section 9, Regular Expressions.

Definition at line 141 of file `regex_constants.h`.

Referenced by `std::regex_traits<_Ch_type>::getloc()`.

4.19.4.8 `constexpr match_flag_type std::regex_constants::format_default`

When a regular expression match is to be replaced by a new string, the new string is constructed using the rules used by the ECMAScript replace function in ECMA- 262 [Ecma International, ECMAScript Language Specification, Standard Ecma-262, third edition, 1999], part 15.5.4.11 String.prototype.replace. In addition, during search and replace operations all non-overlapping occurrences of the regular expression are located and replaced, and sections of the input that did not match the expression are copied unchanged to the output string.

Format strings (from ECMA-262 [15.5.4.11]):

- `$$` The dollar-sign itself (`$`)
- `$&` The matched substring.
- `$'` The portion of *string* that precedes the matched substring. This would be `match_results::prefix()`.
- `$'` The portion of *string* that follows the matched substring. This would be `match_results::suffix()`.
- `$n` The *n*th capture, where *n* is in [1,9] and `$n` is not followed by a decimal digit. If `n <= match_results::size()` and the *n*th capture is undefined, use the empty string instead. If `n > match_results::size()`, the result is implementation-defined.
- `$nn` The *nn*th capture, where *nn* is a two-digit decimal number on [01, 99]. If `nn <= match_results::size()` and the *n*th capture is undefined, use the empty string instead. If `nn > match_results::size()`, the result is implementation-defined.

Definition at line 345 of file `regex_constants.h`.

Referenced by `std::match_results< _Bi_iter >::cend()`.

4.19.4.9 `constexpr match_flag_type std::regex_constants::format_first_only`

When specified during a search and replace operation, only the first occurrence of the regular expression shall be replaced.

Definition at line 368 of file `regex_constants.h`.

Referenced by `std::regex_replace()`.

4.19.4.10 `constexpr match_flag_type std::regex_constants::format_no_copy`

During a search and replace operation, sections of the character container sequence being searched that do not match the regular expression shall not be copied to the output string.

Definition at line 361 of file `regex_constants.h`.

Referenced by `std::regex_replace()`.

4.19.4.11 constexpr match_flag_type std::regex_constants::format_sed

When a regular expression match is to be replaced by a new string, the new string is constructed using the rules used by the POSIX sed utility in IEEE Std 1003.1- 2001 [IEEE, Information Technology – Portable Operating System Interface (POSIX), IEEE Standard 1003.1-2001].

Definition at line 353 of file regex_constants.h.

Referenced by std::regex_traits< _Ch_type >::value().

4.19.4.12 constexpr syntax_option_type std::regex_constants::grep

Specifies that the grammar recognized by the regular expression engine is that used by POSIX utility grep in IEEE Std 1003.1-2001. This option is identical to syntax_option_type basic, except that newlines are treated as whitespace.

Definition at line 161 of file regex_constants.h.

Referenced by std::regex_traits< _Ch_type >::getloc().

4.19.4.13 constexpr syntax_option_type std::regex_constants::icase

Specifies that the matching of regular expressions against a character sequence shall be performed without regard to case.

Definition at line 87 of file regex_constants.h.

Referenced by std::regex_traits< _Ch_type >::getloc().

4.19.4.14 constexpr match_flag_type std::regex_constants::match_any

If more than one match is possible then any match is an acceptable result.

Definition at line 296 of file regex_constants.h.

4.19.4.15 constexpr match_flag_type std::regex_constants::match_continuous

The expression only matches a sub-sequence that begins at first .

Definition at line 308 of file regex_constants.h.

Referenced by std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator++().

4.19.4.16 constexpr match_flag_type std::regex_constants::match_default

The default matching rules.

Definition at line 260 of file regex_constants.h.

Referenced by std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::regex_iterator(), std::regex_match(), std::regex↵_replace(), std::regex_search(), and std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >::regex_token_↵iterator().

4.19.4.17 constexpr match_flag_type std::regex_constants::match_not_bol

The first character in the sequence [first, last) is treated as though it is not at the beginning of a line, so the character (^) in the regular expression shall not match [first, first).

Definition at line 267 of file regex_constants.h.

4.19.4.18 constexpr match_flag_type std::regex_constants::match_not_bow

The expression \b is not matched against the sub-sequence [first,first).

Definition at line 282 of file regex_constants.h.

4.19.4.19 constexpr match_flag_type std::regex_constants::match_not_eol

The last character in the sequence [first, last) is treated as though it is not at the end of a line, so the character (\$) in the regular expression shall not match [last, last).

Definition at line 275 of file regex_constants.h.

4.19.4.20 constexpr match_flag_type std::regex_constants::match_not_eow

The expression \b should not be matched against the sub-sequence [last,last).

Definition at line 289 of file regex_constants.h.

4.19.4.21 constexpr match_flag_type std::regex_constants::match_not_null

The expression does not match an empty sequence.

Definition at line 302 of file regex_constants.h.

Referenced by std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator++().

4.19.4.22 constexpr match_flag_type std::regex_constants::match_prev_avail

—first is a valid iterator position. When this flag is set then the flags match_not_bol and match_not_bow are ignored by the regular expression algorithms 28.11 and iterators 28.12.

Definition at line 316 of file regex_constants.h.

Referenced by std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator++().

4.19.4.23 constexpr syntax_option_type std::regex_constants::nosubs

Specifies that when a regular expression is matched against a character container sequence, no sub-expression matches are to be stored in the supplied match_results structure.

Definition at line 95 of file regex_constants.h.

Referenced by std::regex_traits< _Ch_type >::getloc().

4.19.4.24 constexpr syntax_option_type std::regex_constants::optimize

Specifies that the regular expression engine should pay more attention to the speed with which regular expressions are matched, and less to the speed with which regular expression objects are constructed. Otherwise it has no detectable effect on the program output.

Definition at line 104 of file regex_constants.h.

Referenced by std::regex_traits< _Ch_type >::getloc().

4.20 std::rel_ops Namespace Reference

Functions

- template<class _Tp >
bool **operator!=** (const _Tp &__x, const _Tp &__y)
- template<class _Tp >
bool **operator<=** (const _Tp &__x, const _Tp &__y)
- template<class _Tp >
bool **operator>** (const _Tp &__x, const _Tp &__y)
- template<class _Tp >
bool **operator>=** (const _Tp &__x, const _Tp &__y)

4.20.1 Detailed Description

The generated relational operators are sequestered here.

4.20.2 Function Documentation

4.20.2.1 template<class _Tp > bool std::rel_ops::operator!= (const _Tp &__x, const _Tp &__y) [inline]

Defines != for arbitrary types, in terms of ==.

Parameters

$_x$	A thing.
$_y$	Another thing.

Returns

$_x \text{ != } _y$

This function uses == to determine its result.

Definition at line 87 of file stl_relops.h.

4.20.2.2 `template<class _Tp> bool std::rel_ops::operator<= (const _Tp & __x, const _Tp & __y) [inline]`

Defines `<=` for arbitrary types, in terms of `<`.

Parameters

\longleftrightarrow __x	A thing.
\longleftrightarrow __y	Another thing.

Returns

`__x <= __y`

This function uses `<` to determine its result.

Definition at line 113 of file `stl_relops.h`.

4.20.2.3 `template<class _Tp> bool std::rel_ops::operator> (const _Tp & __x, const _Tp & __y) [inline]`

Defines `>` for arbitrary types, in terms of `<`.

Parameters

\longleftrightarrow __x	A thing.
\longleftrightarrow __y	Another thing.

Returns

`__x > __y`

This function uses `<` to determine its result.

Definition at line 100 of file `stl_relops.h`.

4.20.2.4 `template<class _Tp> bool std::rel_ops::operator>= (const _Tp & __x, const _Tp & __y) [inline]`

Defines `>=` for arbitrary types, in terms of `<`.

Parameters

\longleftrightarrow __x	A thing.
\longleftrightarrow __y	Another thing.

Returns

`__x >= __y`

This function uses `<` to determine its result.

Definition at line 126 of file `stl_relops.h`.

4.21 `std::this_thread` Namespace Reference

Functions

- void `__sleep_for` ([chrono::seconds](#), [chrono::nanoseconds](#))
- [thread::id](#) `get_id` () noexcept
- template<typename `_Rep` , typename `_Period` >
void `sleep_for` (const [chrono::duration](#)< `_Rep`, `_Period` > &`__ptime`)
- template<typename `_Clock` , typename `_Duration` >
void `sleep_until` (const [chrono::time_point](#)< `_Clock`, `_Duration` > &`__ptime`)
- void `yield` () noexcept

4.21.1 Detailed Description

ISO C++ 2011 entities sub-namespace for thread. 30.3.2 Namespace `this_thread`.

4.21.2 Function Documentation

4.21.2.1 `thread::id` `std::this_thread::get_id` () [`inline`], [`noexcept`]

`get_id`

Definition at line 282 of file `thread`.

4.21.2.2 template<typename `_Rep` , typename `_Period` > void `std::this_thread::sleep_for` (const [chrono::duration](#)< `_Rep`, `_Period` > &`__ptime`) [`inline`]

`sleep_for`

Definition at line 310 of file `thread`.

4.21.2.3 template<typename `_Clock` , typename `_Duration` > void `std::this_thread::sleep_until` (const [chrono::time_point](#)< `_Clock`, `_Duration` > &`__ptime`) [`inline`]

`sleep_until`

Definition at line 332 of file `thread`.

4.21.2.4 void std::this_thread::yield () [inline],[noexcept]

yield

Definition at line 297 of file thread.

4.22 std::tr1 Namespace Reference

Namespaces

- [__detail](#)

Functions

- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type [assoc_laguerre](#) (unsigned int __n, unsigned int __m, _Tp __x)
- float [assoc_laguerref](#) (unsigned int __n, unsigned int __m, float __x)
- long double [assoc_laguerrel](#) (unsigned int __n, unsigned int __m, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type [assoc_legendre](#) (unsigned int __l, unsigned int __m, _Tp __x)
- float [assoc_legendref](#) (unsigned int __l, unsigned int __m, float __x)
- long double [assoc_legendrel](#) (unsigned int __l, unsigned int __m, long double __x)
- template<typename _Tpx , typename _Tpy >
__gnu_cxx::__promote_2< _Tpx, _Tpy >::__type [beta](#) (_Tpx __x, _Tpy __y)
- float [betaf](#) (float __x, float __y)
- long double [betal](#) (long double __x, long double __y)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type [comp_ellint_1](#) (_Tp __k)
- float [comp_ellint_1f](#) (float __k)
- long double [comp_ellint_1l](#) (long double __k)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type [comp_ellint_2](#) (_Tp __k)
- float [comp_ellint_2f](#) (float __k)
- long double [comp_ellint_2l](#) (long double __k)
- template<typename _Tp , typename _Tpn >
__gnu_cxx::__promote_2< _Tp, _Tpn >::__type [comp_ellint_3](#) (_Tp __k, _Tpn __nu)
- float [comp_ellint_3f](#) (float __k, float __nu)
- long double [comp_ellint_3l](#) (long double __k, long double __nu)
- template<typename _Tpa , typename _Tpc , typename _Tp >
__gnu_cxx::__promote_3< _Tpa, _Tpc, _Tp >::__type [conf_hyperg](#) (_Tpa __a, _Tpc __c, _Tp __x)
- float [conf_hypergf](#) (float __a, float __c, float __x)
- long double [conf_hypergl](#) (long double __a, long double __c, long double __x)
- template<typename _Tp >
[std::complex](#)< _Tp > [conj](#) (const [std::complex](#)< _Tp > &__z)
- template<typename _Tp >
[std::complex](#)< typename __gnu_cxx::__promote< _Tp >::__type > [conj](#) (_Tp __x)
- template<typename _Tpnu , typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type [cyl_bessel_i](#) (_Tpnu __nu, _Tp __x)
- float [cyl_bessel_if](#) (float __nu, float __x)

- long double **cyl_bessel_il** (long double __nu, long double __x)
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type **cyl_bessel_j** (_Tpnu __nu, _Tp __x)
- float **cyl_bessel_jf** (float __nu, float __x)
- long double **cyl_bessel_jl** (long double __nu, long double __x)
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type **cyl_bessel_k** (_Tpnu __nu, _Tp __x)
- float **cyl_bessel_kf** (float __nu, float __x)
- long double **cyl_bessel_kl** (long double __nu, long double __x)
- template<typename _Tpnu, typename _Tp >
__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type **cyl_neumann** (_Tpnu __nu, _Tp __x)
- float **cyl_neumannf** (float __nu, float __x)
- long double **cyl_neumannl** (long double __nu, long double __x)
- template<typename _Tp, typename _Tpp >
__gnu_cxx::__promote_2< _Tp, _Tpp >::__type **ellint_1** (_Tp __k, _Tpp __phi)
- float **ellint_1f** (float __k, float __phi)
- long double **ellint_1l** (long double __k, long double __phi)
- template<typename _Tp, typename _Tpp >
__gnu_cxx::__promote_2< _Tp, _Tpp >::__type **ellint_2** (_Tp __k, _Tpp __phi)
- float **ellint_2f** (float __k, float __phi)
- long double **ellint_2l** (long double __k, long double __phi)
- template<typename _Tp, typename _Tpn, typename _Tpp >
__gnu_cxx::__promote_3< _Tp, _Tpn, _Tpp >::__type **ellint_3** (_Tp __k, _Tpn __nu, _Tpp __phi)
- float **ellint_3f** (float __k, float __nu, float __phi)
- long double **ellint_3l** (long double __k, long double __nu, long double __phi)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **expint** (_Tp __x)
- float **expintf** (float __x)
- long double **expintl** (long double __x)
- template<typename _Tp >
std::complex< _Tp > **fabs** (const **std::complex**< _Tp > &__z)
- float **fabs** (float __x)
- long double **fabs** (long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **fabs** (_Tp __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **hermite** (unsigned int __n, _Tp __x)
- float **hermitef** (unsigned int __n, float __x)
- long double **hermitel** (unsigned int __n, long double __x)
- template<typename _Tpa, typename _Tpb, typename _Tpc, typename _Tp >
__gnu_cxx::__promote_4< _Tpa, _Tpb, _Tpc, _Tp >::__type **hyperg** (_Tpa __a, _Tpb __b, _Tpc __c, _Tp __x)
- float **hyperg** (float __a, float __b, float __c, float __x)
- long double **hypergl** (long double __a, long double __b, long double __c, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **laguerre** (unsigned int __n, _Tp __x)
- float **laguerref** (unsigned int __n, float __x)
- long double **laguerrel** (unsigned int __n, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **legendre** (unsigned int __n, _Tp __x)
- float **legendref** (unsigned int __n, float __x)
- long double **legendrel** (unsigned int __n, long double __x)

- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > polar (const _Tp &__rho, const _Up`
`&__theta)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > pow (const std::complex< _Tp >`
`&__x, const _Up &__y)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > pow (const _Tp &__x, const std::complex< _Up > &__y)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > pow (const std::complex< _Tp >`
`&__x, const std::complex< _Up > &__y)`
- `template<typename _Tp >`
`std::complex< _Tp > pow (const std::complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp >`
`std::complex< _Tp > pow (const _Tp &__x, const std::complex< _Tp > &__y)`
- `template<typename _Tp >`
`std::complex< _Tp > pow (const std::complex< _Tp > &__x, const std::complex< _Tp > &__y)`
- `float pow (float __x, float __y)`
- `long double pow (long double __x, long double __y)`
- `template<typename _Tp, typename _Up >`
`__gnu_cxx::__promote_2< _Tp, _Up >::__type pow (_Tp __x, _Up __y)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type riemann_zeta (_Tp __x)`
- `float riemann_zetaf (float __x)`
- `long double riemann_zetal (long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type sph_bessel (unsigned int __n, _Tp __x)`
- `float sph_besself (unsigned int __n, float __x)`
- `long double sph_bessell (unsigned int __n, long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type sph_legendre (unsigned int __l, unsigned int __m, _Tp __theta)`
- `float sph_legendref (unsigned int __l, unsigned int __m, float __theta)`
- `long double sph_legendrel (unsigned int __l, unsigned int __m, long double __theta)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type sph_neumann (unsigned int __n, _Tp __x)`
- `float sph_neumannf (unsigned int __n, float __x)`
- `long double sph_neumannl (unsigned int __n, long double __x)`

4.22.1 Detailed Description

ISO C++ TR1 entities toplevel namespace is `std::tr1`.

4.23 `std::tr1::__detail` Namespace Reference

4.23.1 Detailed Description

Implementation details not part of the namespace `std::tr1` interface.

4.24 std::tr2 Namespace Reference

Namespaces

- [__detail](#)

Classes

- struct [__dynamic_bitset_base](#)
- struct [__reflection_typelist](#)
- struct [__reflection_typelist< _First, _Rest... >](#)
- struct [__reflection_typelist<>](#)
- struct [bases](#)
- class [bool_set](#)
- struct [direct_bases](#)
- class [dynamic_bitset](#)

Functions

- bool **certainly** ([bool_set](#) __b)
- bool **contains** ([bool_set](#) __s, [bool_set](#) __t)
- bool **equals** ([bool_set](#) __s, [bool_set](#) __t)
- bool **is_emptyset** ([bool_set](#) __b)
- bool **is_indeterminate** ([bool_set](#) __b)
- bool **is_singleton** ([bool_set](#) __b)
- [bool_set](#) **operator!=** ([bool](#) __s, [bool_set](#) __t)
- [bool_set](#) **operator!=** ([bool_set](#) __s, [bool](#) __t)
- [bool_set](#) **operator!=** ([bool_set](#) __s, [bool_set](#) __t)
- [bool_set](#) **operator&** ([bool](#) __s, [bool_set](#) __t)
- [bool_set](#) **operator&** ([bool_set](#) __s, [bool](#) __t)
- template<typename _CharT, typename _Traits, typename _WordT, typename _Alloc >
[std::basic_ostream](#)< _CharT, _Traits > & **operator<<** ([std::basic_ostream](#)< _CharT, _Traits > &__os, const
[dynamic_bitset](#)< _WordT, _Alloc > &__x)
- [bool_set](#) **operator==** ([bool](#) __s, [bool_set](#) __t)
- [bool_set](#) **operator==** ([bool_set](#) __s, [bool](#) __t)
- template<typename _CharT, typename _Traits, typename _WordT, typename _Alloc >
[std::basic_istream](#)< _CharT, _Traits > & **operator>>** ([std::basic_istream](#)< _CharT, _Traits > &__is, [dynamic_bitset](#)< _WordT, _Alloc > &__x)
- [bool_set](#) **operator^** ([bool](#) __s, [bool_set](#) __t)
- [bool_set](#) **operator^** ([bool_set](#) __s, [bool](#) __t)
- [bool_set](#) **operator|** ([bool](#) __s, [bool_set](#) __t)
- [bool_set](#) **operator|** ([bool_set](#) __s, [bool](#) __t)
- bool **possibly** ([bool_set](#) __b)
- [bool_set](#) **set_complement** ([bool_set](#) __b)
- [bool_set](#) **set_intersection** ([bool](#) __s, [bool_set](#) __t)
- [bool_set](#) **set_intersection** ([bool_set](#) __s, [bool](#) __t)
- [bool_set](#) **set_intersection** ([bool_set](#) __s, [bool_set](#) __t)
- [bool_set](#) **set_union** ([bool](#) __s, [bool_set](#) __t)
- [bool_set](#) **set_union** ([bool_set](#) __s, [bool](#) __t)

- [bool_set set_union](#) ([bool_set](#) __s, [bool_set](#) __t)
- `template<typename _WordT, typename _Alloc >`
`bool operator!= (const dynamic_bitset< _WordT, _Alloc > &__lhs, const dynamic_bitset< _WordT, _Alloc >`
`&__rhs)`
- `template<typename _WordT, typename _Alloc >`
`bool operator<= (const dynamic_bitset< _WordT, _Alloc > &__lhs, const dynamic_bitset< _WordT, _Alloc >`
`&__rhs)`
- `template<typename _WordT, typename _Alloc >`
`bool operator> (const dynamic_bitset< _WordT, _Alloc > &__lhs, const dynamic_bitset< _WordT, _Alloc >`
`&__rhs)`
- `template<typename _WordT, typename _Alloc >`
`bool operator>= (const dynamic_bitset< _WordT, _Alloc > &__lhs, const dynamic_bitset< _WordT, _Alloc >`
`&__rhs)`
- `template<typename _WordT, typename _Alloc >`
`dynamic_bitset< _WordT, _Alloc > operator& (const dynamic_bitset< _WordT, _Alloc > &__x, const dynamic_bitset< _WordT, _Alloc > &__y)`
- `template<typename _WordT, typename _Alloc >`
`dynamic_bitset< _WordT, _Alloc > operator| (const dynamic_bitset< _WordT, _Alloc > &__x, const dynamic_bitset< _WordT, _Alloc > &__y)`
- `template<typename _WordT, typename _Alloc >`
`dynamic_bitset< _WordT, _Alloc > operator^ (const dynamic_bitset< _WordT, _Alloc > &__x, const dynamic_bitset< _WordT, _Alloc > &__y)`
- `template<typename _WordT, typename _Alloc >`
`dynamic_bitset< _WordT, _Alloc > operator- (const dynamic_bitset< _WordT, _Alloc > &__x, const dynamic_bitset< _WordT, _Alloc > &__y)`

4.24.1 Detailed Description

ISO C++ TR2 entities toplevel namespace is `std::tr2`.

4.25 `std::tr2::__detail` Namespace Reference

4.25.1 Detailed Description

Implementation details not part of the namespace `std::tr2` interface.

5 Class Documentation

5.1 `__cxxabiv1::__forced_unwind` Class Reference

5.1.1 Detailed Description

Thrown as part of forced unwinding.

A magic placeholder class that can be caught by reference to recognize forced unwinding.

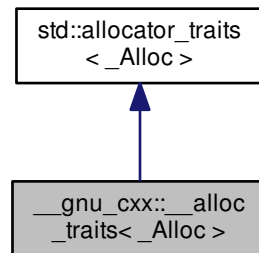
Definition at line 48 of file `cxxabi_forced.h`.

The documentation for this class was generated from the following file:

- [cxxabi_forced.h](#)

5.2 `__gnu_cxx::__alloc_traits<_Alloc>` Struct Template Reference

Inheritance diagram for `__gnu_cxx::__alloc_traits<_Alloc>`:



Public Types

- `template<typename _Alloc, typename _Up>`
`using __rebind = typename _Alloc::template rebind<_Up>::other`
- `typedef std::allocator_traits<_Alloc> _Base_type`
- `typedef _Alloc allocator_type`
- `typedef _Base_type::const_pointer const_pointer`
- `typedef const value_type & const_reference`
- `using const_void_pointer = __detected_or_t<__ptr_rebind<pointer, const void>, __cv_pointer, _Alloc>`
- `typedef _Base_type::difference_type difference_type`
- `using is_always_equal = __detected_or_t<typename is_empty<_Alloc>::type, __equal, _Alloc>`
- `typedef _Base_type::pointer pointer`
- `using propagate_on_container_copy_assignment = __detected_or_t<>false_type, __pocca, _Alloc>`
- `using propagate_on_container_move_assignment = __detected_or_t<>false_type, __pocma, _Alloc>`
- `using propagate_on_container_swap = __detected_or_t<>false_type, __pocs, _Alloc>`
- `template<typename _Tp>`
`using rebind_alloc = __alloc_rebind<_Alloc, _Tp>`
- `template<typename _Tp>`
`using rebind_traits = allocator_traits<rebind_alloc<_Tp>>`
- `typedef value_type & reference`
- `typedef _Base_type::size_type size_type`
- `typedef _Base_type::value_type value_type`
- `using void_pointer = __detected_or_t<__ptr_rebind<pointer, void>, __v_pointer, _Alloc>`

Static Public Member Functions

- static constexpr bool **_S_always_equal** ()
- static constexpr bool **_S_nothrow_move** ()
- static void **_S_on_swap** (_Alloc &__a, _Alloc &__b)
- static constexpr bool **_S_propagate_on_copy_assign** ()
- static constexpr bool **_S_propagate_on_move_assign** ()
- static constexpr bool **_S_propagate_on_swap** ()
- static _Alloc **_S_select_on_copy** (const _Alloc &__a)
- static pointer allocate (_Alloc &__a, size_type __n)
- static pointer allocate (_Alloc &__a, size_type __n, const_void_pointer __hint)
- template<typename _Ptr, typename... _Args>
static std::enable_if< __is_custom_pointer< _Ptr >::value >::type **construct** (_Alloc &__a, _Ptr __p, _Args &&... __args)
- template<typename _Tp, typename... _Args>
static auto **construct** (_Alloc &__a, _Tp *__p, _Args &&... __args) -> decltype(_S_construct(__a, __p, std::forward< _Args >(__args)...))
- static void **deallocate** (_Alloc &__a, pointer __p, size_type __n)
- template<typename _Ptr >
static std::enable_if< __is_custom_pointer< _Ptr >::value >::type **destroy** (_Alloc &__a, _Ptr __p)
- template<typename _Tp >
static void **destroy** (_Alloc &__a, _Tp *__p)
- static size_type **max_size** (const _Alloc &__a) noexcept
- static _Alloc **select_on_container_copy_construction** (const _Alloc &__rhs)

Protected Types

- template<typename _Tp >
using **__c_pointer** = typename _Tp::const_pointer
- template<typename _Tp >
using **__cv_pointer** = typename _Tp::const_void_pointer
- template<typename _Tp >
using **__diff_type** = typename _Tp::difference_type
- template<typename _Tp >
using **__equal** = typename _Tp::is_always_equal
- template<typename _Tp >
using **__pocca** = typename _Tp::propagate_on_container_copy_assignment
- template<typename _Tp >
using **__pocma** = typename _Tp::propagate_on_container_move_assignment
- template<typename _Tp >
using **__pocs** = typename _Tp::propagate_on_container_swap
- template<typename _Tp >
using **__pointer** = typename _Tp::pointer
- template<typename _Tp >
using **__size_type** = typename _Tp::size_type
- template<typename _Tp >
using **__v_pointer** = typename _Tp::void_pointer

5.2.1 Detailed Description

```
template<typename _Alloc>
struct __gnu_cxx::__alloc_traits<_Alloc>
```

Uniform interface to C++98 and C++11 allocators.

Definition at line 50 of file `ext/alloc_traits.h`.

5.2.2 Member Typedef Documentation

5.2.2.1 `template<typename _Alloc> using std::allocator_traits<_Alloc>::const_void_pointer = __detected_or_t<__ptr_rebind<pointer, const void>, __cv_pointer, _Alloc> [inherited]`

The allocator's const void pointer type.

`Alloc::const_void_pointer` if that type exists, otherwise `pointer_traits<pointer>::rebind<const void>`

Definition at line 124 of file `bits/alloc_traits.h`.

5.2.2.2 `template<typename _Alloc> using std::allocator_traits<_Alloc>::is_always_equal = __detected_or_t<typename is_empty<_Alloc>::type, __equal, _Alloc> [inherited]`

Whether all instances of the allocator type compare equal.

`Alloc::is_always_equal` if that type exists, otherwise `is_empty<Alloc>::type`

Definition at line 180 of file `bits/alloc_traits.h`.

5.2.2.3 `template<typename _Alloc> using std::allocator_traits<_Alloc>::propagate_on_container_copy_assignment = __detected_or_t<false_type, __pocca, _Alloc> [inherited]`

How the allocator is propagated on copy assignment.

`Alloc::propagate_on_container_copy_assignment` if that type exists, otherwise `false_type`

Definition at line 153 of file `bits/alloc_traits.h`.

5.2.2.4 `template<typename _Alloc> using std::allocator_traits<_Alloc>::propagate_on_container_move_assignment = __detected_or_t<false_type, __pocma, _Alloc> [inherited]`

How the allocator is propagated on move assignment.

`Alloc::propagate_on_container_move_assignment` if that type exists, otherwise `false_type`

Definition at line 162 of file `bits/alloc_traits.h`.

5.2.2.5 `template<typename _Alloc> using std::allocator_traits< _Alloc >::propagate_on_container_swap =
__detected_or_t<false_type, __pocs, _Alloc> [inherited]`

How the allocator is propagated on swap.

`Alloc::propagate_on_container_swap` if that type exists, otherwise `false_type`

Definition at line 171 of file `bits/alloc_traits.h`.

5.2.2.6 `template<typename _Alloc> using std::allocator_traits< _Alloc >::void_pointer =
__detected_or_t<__ptr_rebind<pointer, void>, __v_pointer, _Alloc> [inherited]`

The allocator's void pointer type.

`Alloc::void_pointer` if that type exists, otherwise `pointer_traits<pointer>::rebind<void>`

Definition at line 114 of file `bits/alloc_traits.h`.

5.2.3 Member Function Documentation

5.2.3.1 `template<typename _Alloc> static pointer std::allocator_traits< _Alloc >::allocate (_Alloc & __a, size_type __n)
[inline], [static], [inherited]`

Allocate memory.

Parameters

<code>__a</code>	An allocator.
<code>__n</code>	The number of objects to allocate space for.

Calls `a.allocate(n)`

Definition at line 280 of file `bits/alloc_traits.h`.

Referenced by `std::__allocate_guarded()`.

5.2.3.2 `template<typename _Alloc> static pointer std::allocator_traits< _Alloc >::allocate (_Alloc & __a, size_type __n,
const_void_pointer __hint) [inline], [static], [inherited]`

Allocate memory.

Parameters

<code>__a</code>	An allocator.
<code>__n</code>	The number of objects to allocate space for.
<code>__hint</code>	Aid to locality.

Returns

Memory of suitable size and alignment for n objects of type `value_type`

Returns `a.allocate(n, hint)` if that expression is well-formed, otherwise returns `a.allocate(n)`

Definition at line 295 of file `bits/alloc_traits.h`.

5.2.3.3 `template<typename _Alloc> template<typename _Tp, typename... _Args> static auto std::allocator_traits<_Alloc>::construct (_Alloc & __a, _Tp * __p, _Args &&... __args) -> decltype(_S_construct(__a, __p, std::forward<_Args>(__args)...))` `[inline], [static], [inherited]`

Construct an object of type `_Tp`.

Parameters

<code>__a</code>	An allocator.
<code>__p</code>	Pointer to memory of suitable size and alignment for <code>Tp</code>
<code>__args</code>	Constructor arguments.

Calls `__a.construct(__p, std::forward<Args>(__args)...)` if that expression is well-formed, otherwise uses placement-new to construct an object of type `_Tp` at location `__p` from the arguments `__args...`

Definition at line 322 of file `bits/alloc_traits.h`.

5.2.3.4 `template<typename _Alloc> static void std::allocator_traits<_Alloc>::deallocate (_Alloc & __a, pointer __p, size_type __n)` `[inline], [static], [inherited]`

Deallocate memory.

Parameters

<code>__a</code>	An allocator.
<code>__p</code>	Pointer to the memory to deallocate.
<code>__n</code>	The number of objects space was allocated for.

Calls `a.deallocate(p, n)`

Definition at line 307 of file `bits/alloc_traits.h`.

Referenced by `std::__allocated_ptr<_Alloc>::~~__allocated_ptr()`.

5.2.3.5 `template<typename _Alloc> template<typename _Tp> static void std::allocator_traits<_Alloc>::destroy (_Alloc & __a, _Tp * __p)` `[inline], [static], [inherited]`

Destroy an object of type `_Tp`.

Parameters

$_a$	An allocator.
$_p$	Pointer to the object to destroy

Calls `__a.destroy(__p)` if that expression is well-formed, otherwise calls `__p->~Tp()`

Definition at line 335 of file `bits/alloc_traits.h`.

5.2.3.6 `template<typename _Alloc> static size_type std::allocator_traits<_Alloc>::max_size (const _Alloc & __a)`
`[inline], [static], [noexcept], [inherited]`

The maximum supported allocation size.

Parameters

$_a$	An allocator.
-------	---------------

Returns

`__a.max_size()` or `numeric_limits<size_type>::max()`

Returns `__a.max_size()` if that expression is well-formed, otherwise returns `numeric_limits<size_type>::max()`

Definition at line 346 of file `bits/alloc_traits.h`.

5.2.3.7 `template<typename _Alloc> static _Alloc std::allocator_traits<_Alloc>::select_on_container_copy_construction (`
`const _Alloc & __rhs) [inline], [static], [inherited]`

Obtain an allocator to use when copying a container.

Parameters

$_rhs$	An allocator.
---------	---------------

Returns

`__rhs.select_on_container_copy_construction()` or `__rhs`

Returns `__rhs.select_on_container_copy_construction()` if that expression is well-formed, otherwise returns `__rhs`

Definition at line 358 of file `bits/alloc_traits.h`.

The documentation for this struct was generated from the following file:

- [ext/alloc_traits.h](#)

5.3 `__gnu_cxx::__common_pool_policy<_PoolTp, _Thread >` Struct Template Reference

Inherits `__gnu_cxx::__common_pool_base<_PoolTp, _Thread >`.

5.3.1 Detailed Description

```
template<template< bool > class _PoolTp, bool _Thread>
struct __gnu_cxx::__common_pool_policy<_PoolTp, _Thread >
```

Policy for shared `__pool` objects.

Definition at line 460 of file `mt_allocator.h`.

The documentation for this struct was generated from the following file:

- [mt_allocator.h](#)

5.4 `__gnu_cxx::__detail::__mini_vector<_Tp >` Class Template Reference

Public Types

- typedef const `_Tp` & **const_reference**
- typedef ptrdiff_t **difference_type**
- typedef pointer **iterator**
- typedef `_Tp *` **pointer**
- typedef `_Tp &` **reference**
- typedef size_t **size_type**
- typedef `_Tp` **value_type**

Public Member Functions

- reference **back** () const throw ()
- iterator **begin** () const throw ()
- void **clear** () throw ()
- iterator **end** () const throw ()
- void **erase** (iterator __pos) throw ()
- void **insert** (iterator __pos, const_reference __x)
- reference **operator[]** (const size_type __pos) const throw ()
- void **pop_back** () throw ()
- void **push_back** (const_reference __x)
- size_type **size** () const throw ()

5.4.1 Detailed Description

```
template<typename _Tp>
class __gnu_cxx::__detail::__mini_vector< _Tp >
```

`__mini_vector<>` is a stripped down version of the full-fledged `std::vector<>`.

It is to be used only for built-in types or PODs. Notable differences are:

1. Not all accessor functions are present. 2. Used ONLY for PODs. 3. No Allocator template argument. Uses operator `new()` to get memory, and operator `delete()` to free it. Caveat: The dtor does NOT free the memory allocated, so this a memory-leaking vector!

Definition at line 69 of file `bitmap_allocator.h`.

The documentation for this class was generated from the following file:

- [bitmap_allocator.h](#)

5.5 `__gnu_cxx::__detail::Bitmap_counter< _Tp >` Class Template Reference

Public Member Functions

- `_Bitmap_counter` ([_BPVector](#) &Rvbp, long __index=-1)
- pointer `_M_base` () const throw ()
- bool `_M_finished` () const throw ()
- `size_t * _M_get` () const throw ()
- `_Index_type _M_offset` () const throw ()
- void `_M_reset` (long __index=-1) throw ()
- void `_M_set_internal_bitmap` (`size_t * __new_internal_marker`) throw ()
- `_Index_type _M_where` () const throw ()
- `_Bitmap_counter` & `operator++` () throw ()

5.5.1 Detailed Description

```
template<typename _Tp>
class __gnu_cxx::__detail::Bitmap_counter< _Tp >
```

The bitmap counter which acts as the bitmap manipulator, and manages the bit-manipulation functions and the searching and identification functions on the bit-map.

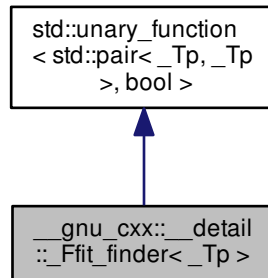
Definition at line 396 of file `bitmap_allocator.h`.

The documentation for this class was generated from the following file:

- [bitmap_allocator.h](#)

5.6 `__gnu_cxx::__detail::_Ffit_finder<_Tp>` Class Template Reference

Inheritance diagram for `__gnu_cxx::__detail::_Ffit_finder<_Tp>`:



Public Types

- typedef `std::pair<_Tp, _Tp>` `argument_type`
- typedef `bool` `result_type`

Public Member Functions

- `size_t * _M_get () const throw ()`
- `_Counter_type _M_offset () const throw ()`
- `bool operator() (_Block_pair __bp) throw ()`

5.6.1 Detailed Description

```
template<typename _Tp>
class __gnu_cxx::__detail::_Ffit_finder<_Tp>
```

The class which acts as a predicate for applying the first-fit memory allocation policy for the bitmap allocator.

Definition at line 331 of file `bitmap_allocator.h`.

5.6.2 Member Typedef Documentation

5.6.2.1 typedef `std::pair<_Tp, _Tp>` `std::unary_function<std::pair<_Tp, _Tp>, bool>::argument_type` [inherited]

`argument_type` is the type of the argument

Definition at line 108 of file `stl_function.h`.

5.6.2.2 `typedef bool std::unary_function< std::pair<_Tp,_Tp>,bool>::result_type` [inherited]

`result_type` is the return type

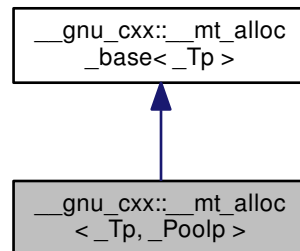
Definition at line 111 of file `stl_function.h`.

The documentation for this class was generated from the following file:

- [bitmap_allocator.h](#)

5.7 `__gnu_cxx::__mt_alloc<_Tp,_Poolp>` Class Template Reference

Inheritance diagram for `__gnu_cxx::__mt_alloc<_Tp,_Poolp>`:



Public Types

- `typedef _Poolp __policy_type`
- `typedef _Poolp::pool_type __pool_type`
- `typedef const _Tp * const_pointer`
- `typedef const _Tp & const_reference`
- `typedef ptrdiff_t difference_type`
- `typedef _Tp * pointer`
- `typedef std::true_type propagate_on_container_move_assignment`
- `typedef _Tp & reference`
- `typedef size_t size_type`
- `typedef _Tp value_type`

Public Member Functions

- `__mt_alloc` (const `__mt_alloc` &) noexcept
- `template<typename _Tp1, typename _Poolp1 >`
`__mt_alloc` (const `__mt_alloc<_Tp1, _Poolp1>` &) noexcept
- `const __pool_base::Tune __M_get_options` ()
- `void __M_set_options` (`__pool_base::Tune __t`)
- `pointer address` (reference `__x`) const noexcept
- `const_pointer address` (const_reference `__x`) const noexcept
- `pointer allocate` (size_type `__n`, const void `*=0`)
- `template<typename _Up, typename... _Args>`
`void construct` (`_Up *__p, _Args &&... __args`)
- `void deallocate` (pointer `__p`, size_type `__n`)
- `template<typename _Up >`
`void destroy` (`_Up *__p`)
- size_type `max_size` () const noexcept

5.7.1 Detailed Description

```
template<typename _Tp, typename _Poolp = __common_pool_policy<__pool, true >>
class __gnu_cxx::__mt_alloc<_Tp, _Poolp>
```

This is a fixed size (power of 2) allocator which - when compiled with thread support - will maintain one freelist per size per thread plus a *global* one. Steps are taken to limit the per thread freelist sizes (by returning excess back to the *global* list).

Further details: https://gcc.gnu.org/onlinedocs/libstdc++/manual/mt_allocator.html.

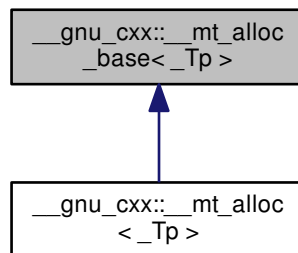
Definition at line 639 of file `mt_allocator.h`.

The documentation for this class was generated from the following file:

- [mt_allocator.h](#)

5.8 `__gnu_cxx::__mt_alloc_base<_Tp>` Class Template Reference

Inheritance diagram for `__gnu_cxx::__mt_alloc_base<_Tp>`:



Public Types

- typedef const _Tp * **const_pointer**
- typedef const _Tp & **const_reference**
- typedef ptrdiff_t **difference_type**
- typedef _Tp * **pointer**
- typedef [std::true_type](#) **propagate_on_container_move_assignment**
- typedef _Tp & **reference**
- typedef size_t **size_type**
- typedef _Tp **value_type**

Public Member Functions

- pointer **address** (reference __x) const noexcept
- const_pointer **address** (const_reference __x) const noexcept
- template<typename _Up, typename... _Args>
void **construct** (_Up *__p, _Args &&... __args)
- template<typename _Up >
void **destroy** (_Up *__p)
- size_type **max_size** () const noexcept

5.8.1 Detailed Description

```
template<typename _Tp>
class __gnu_cxx::__mt_alloc_base<_Tp>
```

Base class for _Tp dependent member functions.

Definition at line 570 of file mt_allocator.h.

The documentation for this class was generated from the following file:

- [mt_allocator.h](#)

5.9 __gnu_cxx::__per_type_pool_policy<_Tp, _PoolTp, _Thread> Struct Template Reference

Inherits [__gnu_cxx::__per_type_pool_base<_Tp, _PoolTp, _Thread>](#).

5.9.1 Detailed Description

```
template<typename _Tp, template< bool > class _PoolTp, bool _Thread>
struct __gnu_cxx::__per_type_pool_policy<_Tp, _PoolTp, _Thread>
```

Policy for individual __pool objects.

Definition at line 555 of file mt_allocator.h.

The documentation for this struct was generated from the following file:

- [mt_allocator.h](#)

5.10 __gnu_cxx::__pool<_Thread> Class Template Reference

5.10.1 Detailed Description

```
template<bool _Thread>
class __gnu_cxx::__pool<_Thread>
```

Data describing the underlying memory pool, parameterized on threading support.

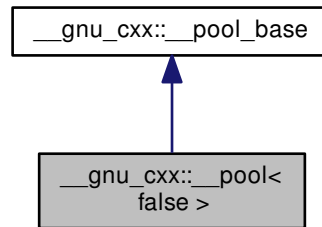
Definition at line 192 of file `mt_allocator.h`.

The documentation for this class was generated from the following file:

- [mt_allocator.h](#)

5.11 __gnu_cxx::__pool<false> Class Template Reference

Inheritance diagram for `__gnu_cxx::__pool<false>`:



Public Types

- typedef unsigned short int **_Binmap_type**

Public Member Functions

- **__pool** (const `__pool_base::Tune &` `__tune`)
- void **_M_adjust_freelist** (const `_Bin_record &`, `_Block_record *`, `size_t`)
- bool **_M_check_threshold** (`size_t` `__bytes`)
- void **_M_destroy** () throw ()
- `size_t` **_M_get_align** ()
- const `_Bin_record &` **_M_get_bin** (`size_t` `__which`)
- `size_t` **_M_get_binmap** (`size_t` `__bytes`)
- const `_Tune &` **_M_get_options** () const
- `size_t` **_M_get_thread_id** ()
- void **_M_initialize_once** ()
- void **_M_reclaim_block** (`char *` `__p`, `size_t` `__bytes`) throw ()
- `char *` **_M_reserve_block** (`size_t` `__bytes`, const `size_t` `__thread_id`)
- void **_M_set_options** (`_Tune` `__t`)

Protected Attributes

- `_Binmap_type * _M_binmap`
- `bool _M_init`
- `_Tune _M_options`

5.11.1 Detailed Description

```
template<>
class __gnu_cxx::__pool< false >
```

Specialization for single thread.

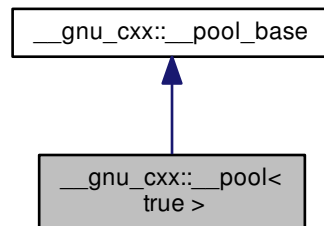
Definition at line 196 of file `mt_allocator.h`.

The documentation for this class was generated from the following file:

- [mt_allocator.h](#)

5.12 __gnu_cxx::__pool< true > Class Template Reference

Inheritance diagram for `__gnu_cxx::__pool< true >`:



Public Types

- `typedef unsigned short int _Binmap_type`

Public Member Functions

- **__pool** (const __pool_base::__Tune &__tune)
- void **_M_adjust_freelist** (const _Bin_record &__bin, _Block_record *__block, size_t __thread_id)
- bool **_M_check_threshold** (size_t __bytes)
- void **_M_destroy** () throw ()
- void **_M_destroy_thread_key** (void *) throw ()
- size_t **_M_get_align** ()
- const _Bin_record & **_M_get_bin** (size_t __which)
- size_t **_M_get_binmap** (size_t __bytes)
- const _Tune & **_M_get_options** () const
- size_t **_M_get_thread_id** ()
- void **_M_initialize** (__destroy_handler)
- void **_M_initialize_once** ()
- void **_M_reclaim_block** (char *__p, size_t __bytes) throw ()
- char * **_M_reserve_block** (size_t __bytes, const size_t __thread_id)
- void **_M_set_options** (_Tune __t)

Protected Attributes

- _Binmap_type * **_M_binmap**
- bool **_M_init**
- _Tune **_M_options**

5.12.1 Detailed Description

```
template<>
```

```
class __gnu_cxx::__pool< true >
```

Specialization for thread enabled, via gthreads.h.

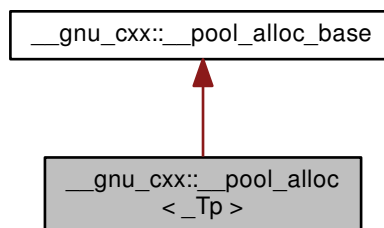
Definition at line 263 of file mt_allocator.h.

The documentation for this class was generated from the following file:

- [mt_allocator.h](#)

5.13 __gnu_cxx::__pool_alloc<_Tp> Class Template Reference

Inheritance diagram for __gnu_cxx::__pool_alloc<_Tp>:



Public Types

- typedef const _Tp * **const_pointer**
- typedef const _Tp & **const_reference**
- typedef ptrdiff_t **difference_type**
- typedef _Tp * **pointer**
- typedef [std::true_type](#) **propagate_on_container_move_assignment**
- typedef _Tp & **reference**
- typedef size_t **size_type**
- typedef _Tp **value_type**

Public Member Functions

- **__pool_alloc** (const [__pool_alloc](#) &) noexcept
- template<typename _Tp1 >
 __pool_alloc (const [__pool_alloc](#)<_Tp1 > &) noexcept
- pointer **address** (reference __x) const noexcept
- const_pointer **address** (const_reference __x) const noexcept
- pointer **allocate** (size_type __n, const void *=0)
- template<typename _Up, typename... _Args>
 void **construct** (_Up * __p, _Args &&... __args)
- void **deallocate** (pointer __p, size_type __n)
- template<typename _Up >
 void **destroy** (_Up * __p)
- size_type **max_size** () const noexcept

Private Types

- enum { **_S_align** }
- enum { **_S_max_bytes** }
- enum { **_S_free_list_size** }

Private Member Functions

- char * **_M_allocate_chunk** (size_t __n, int & __nobjs)
- _Obj *volatile * **_M_get_free_list** (size_t __bytes) throw ()
- __mutex & **_M_get_mutex** () throw ()
- void * **_M_refill** (size_t __n)
- size_t **_M_round_up** (size_t __bytes)

Static Private Attributes

- static char * **_S_end_free**
- static _Obj *volatile **_S_free_list** [_S_free_list_size]
- static size_t **_S_heap_size**
- static char * **_S_start_free**

5.13.1 Detailed Description

```
template<typename _Tp>
class __gnu_cxx::__pool_alloc<_Tp>
```

Allocator using a memory pool with a single lock.

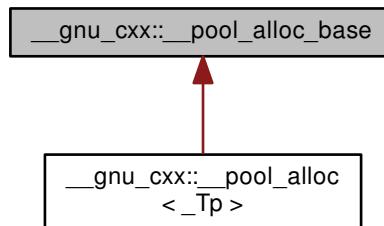
Definition at line 126 of file pool_allocator.h.

The documentation for this class was generated from the following file:

- [pool_allocator.h](#)

5.14 __gnu_cxx::__pool_alloc_base Class Reference

Inheritance diagram for __gnu_cxx::__pool_alloc_base:



Protected Types

- enum { **_S_align** }
- enum { **_S_max_bytes** }
- enum { **_S_free_list_size** }

Protected Member Functions

- char * **_M_allocate_chunk** (size_t __n, int &__nobjs)
- _Obj *volatile * **_M_get_free_list** (size_t __bytes) throw ()
- __mutex & **_M_get_mutex** () throw ()
- void * **_M_refill** (size_t __n)
- size_t **_M_round_up** (size_t __bytes)

Static Protected Attributes

- static char * **_S_end_free**
- static _Obj *volatile **_S_free_list** [_S_free_list_size]
- static size_t **_S_heap_size**
- static char * **_S_start_free**

5.14.1 Detailed Description

Base class for `__pool_alloc`.

Uses various allocators to fulfill underlying requests (and makes as few requests as possible when in default high-speed pool mode).

Important implementation properties: 0. If globally mandated, then allocate objects from new 1. If the clients request an object of size $> _S_max_bytes$, the resulting object will be obtained directly from new 2. In all other cases, we allocate an object of size exactly $_S_round_up(requested_size)$. Thus the client has enough size information that we can return the object to the proper free list without permanently losing part of the object.

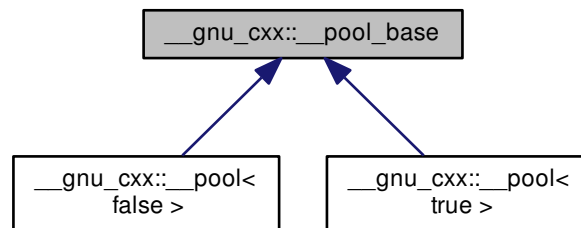
Definition at line 78 of file `pool_allocator.h`.

The documentation for this class was generated from the following file:

- [pool_allocator.h](#)

5.15 `__gnu_cxx::__pool_base` Struct Reference

Inheritance diagram for `__gnu_cxx::__pool_base`:



Public Types

- typedef unsigned short int **_Binmap_type**

Public Member Functions

- `__pool_base` (const `_Tune` & `__options`)
- `bool _M_check_threshold` (size_t `__bytes`)
- `size_t _M_get_align` ()
- `size_t _M_get_binmap` (size_t `__bytes`)
- `const _Tune & _M_get_options` () const
- `void _M_set_options` (`_Tune` `__t`)

Protected Attributes

- `_Binmap_type * _M_binmap`
- `bool _M_init`
- `_Tune _M_options`

5.15.1 Detailed Description

Base class for pool object.

Definition at line 51 of file `mt_allocator.h`.

The documentation for this struct was generated from the following file:

- [mt_allocator.h](#)

5.16 `__gnu_cxx::__rc_string_base< _CharT, _Traits, _Alloc >` Class Template Reference

Inherits `__gnu_cxx::__vstring_utility< _CharT, _Traits, _Alloc >`.

Public Types

- `typedef _Util_Base::_CharT_alloc_type _CharT_alloc_type`
- `typedef __vstring_utility< _CharT, _Traits, _Alloc > _Util_Base`
- `typedef _Alloc allocator_type`
- `typedef _CharT_alloc_type::size_type size_type`
- `typedef _Traits traits_type`
- `typedef _Traits::char_type value_type`

Public Member Functions

- `__rc_string_base` (const `_Alloc` & `_a`)
- `__rc_string_base` (const `__rc_string_base` & `__rcs`)
- `__rc_string_base` (`__rc_string_base` && `__rcs`)
- `__rc_string_base` (size_type `__n`, `_CharT` `__c`, const `_Alloc` & `_a`)
- template<typename `_InputIterator` >
`__rc_string_base` (`_InputIterator` `__beg`, `_InputIterator` `__end`, const `_Alloc` & `_a`)
- void `_M_assign` (const `__rc_string_base` & `__rcs`)
- size_type `_M_capacity` () const
- void `_M_clear` ()
- bool `_M_compare` (const `__rc_string_base` &) const
- template<>
bool `_M_compare` (const `__rc_string_base` & `__rcs`) const
- template<>
bool `_M_compare` (const `__rc_string_base` & `__rcs`) const
- `_CharT` * `_M_data` () const
- void `_M_erase` (size_type `__pos`, size_type `__n`)
- allocator_type & `_M_get_allocator` ()
- const allocator_type & `_M_get_allocator` () const
- bool `_M_is_shared` () const
- void `_M_leak` ()
- size_type `_M_length` () const
- size_type `_M_max_size` () const
- void `_M_mutate` (size_type `__pos`, size_type `__len1`, const `_CharT` * `__s`, size_type `__len2`)
- void `_M_reserve` (size_type `__res`)
- void `_M_set_leaked` ()
- void `_M_set_length` (size_type `__n`)
- void `_M_swap` (`__rc_string_base` & `__rcs`)
- template<typename `_InIterator` >
`_CharT` * `_S_construct` (`_InIterator` `__beg`, `_InIterator` `__end`, const `_Alloc` & `_a`, `std::forward_iterator_tag`)

Protected Types

- typedef `__gnu_cxx::__normal_iterator`< const_pointer, `__gnu_cxx::__versa_string`< `_CharT`, `_Traits`, `_Alloc`, `__rc_string_base` > > `__const_rc_iterator`
- typedef `__gnu_cxx::__normal_iterator`< const_pointer, `__gnu_cxx::__versa_string`< `_CharT`, `_Traits`, `_Alloc`, `__sso_string_base` > > `__const_sso_iterator`
- typedef `__gnu_cxx::__normal_iterator`< pointer, `__gnu_cxx::__versa_string`< `_CharT`, `_Traits`, `_Alloc`, `__rc_string_base` > > `__rc_iterator`
- typedef `__gnu_cxx::__normal_iterator`< pointer, `__gnu_cxx::__versa_string`< `_CharT`, `_Traits`, `_Alloc`, `__sso_string_base` > > `__sso_iterator`
- typedef `_CharT_alloc_type::const_pointer` `const_pointer`
- typedef `_CharT_alloc_type::difference_type` `difference_type`
- typedef `_CharT_alloc_type::pointer` `pointer`

Static Protected Member Functions

- static void `_S_assign` (`_CharT *__d`, `size_type __n`, `_CharT __c`)
- static int `_S_compare` (`size_type __n1`, `size_type __n2`)
- static void `_S_copy` (`_CharT *__d`, `const _CharT *__s`, `size_type __n`)
- template<typename `_Iterator`>
static void `_S_copy_chars` (`_CharT *__p`, `_Iterator __k1`, `_Iterator __k2`)
- static void `_S_copy_chars` (`_CharT *__p`, `__sso_iterator __k1`, `__sso_iterator __k2`)
- static void `_S_copy_chars` (`_CharT *__p`, `__const_sso_iterator __k1`, `__const_sso_iterator __k2`)
- static void `_S_copy_chars` (`_CharT *__p`, `__rc_iterator __k1`, `__rc_iterator __k2`)
- static void `_S_copy_chars` (`_CharT *__p`, `__const_rc_iterator __k1`, `__const_rc_iterator __k2`)
- static void `_S_copy_chars` (`_CharT *__p`, `_CharT *__k1`, `_CharT *__k2`)
- static void `_S_copy_chars` (`_CharT *__p`, `const _CharT *__k1`, `const _CharT *__k2`)
- static void `_S_move` (`_CharT *__d`, `const _CharT *__s`, `size_type __n`)

5.16.1 Detailed Description

```
template<typename _CharT, typename _Traits, typename _Alloc>
class __gnu_cxx::__rc_string_base<_CharT, _Traits, _Alloc>
```

Documentation? What's that? Nathan Myers ncm@cantrip.org.

A string looks like this:

```

                                     [_Rep]
                                     _M_length
[ __rc_string_base<char_type>]      _M_capacity
_M_dataplus                       _M_refcount
_M_p ----->                     unnamed array of char_type
```

Where the `_M_p` points to the first character in the string, and you cast it to a pointer-to-`_Rep` and subtract 1 to get a pointer to the header.

This approach has the enormous advantage that a string object requires only one allocation. All the ugliness is confined within a single pair of inline functions, which each compile to a single *add* instruction: `_Rep::_M_refdata()`, and `__rc_string_base::_M_rep()`; and the allocation function which gets a block of raw bytes and with room enough and constructs a `_Rep` object at the front.

The reason you want `_M_data` pointing to the character array and not the `_Rep` is so that the debugger can see the string contents. (Probably we should add a non-inline member to get the `_Rep` for the debugger to use, so users can check the actual string length.)

Note that the `_Rep` object is a POD so that you can have a static *empty string* `_Rep` object already *constructed* before static constructors have run. The reference-count encoding is chosen so that a 0 indicates one reference, so you never try to destroy the empty-string `_Rep` object.

All but the last paragraph is considered pretty conventional for a C++ string implementation.

Definition at line 82 of file `rc_string_base.h`.

The documentation for this class was generated from the following file:

- [rc_string_base.h](#)

5.17 `__gnu_cxx::__scoped_lock` Class Reference

Public Types

- typedef `__mutex` `__mutex_type`

Public Member Functions

- `__scoped_lock` (`__mutex_type` &`__name`)

5.17.1 Detailed Description

Scoped lock idiom.

Definition at line 231 of file `concurrency.h`.

The documentation for this class was generated from the following file:

- [concurrency.h](#)

5.18 `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >` Class Template Reference

Inherits `_Base<_CharT, _Traits, _Alloc >`.

Public Types

- typedef `_Alloc` `allocator_type`
- typedef `__gnu_cxx::__normal_iterator< const_pointer, __versa_string >` `const_iterator`
- typedef `_CharT_alloc_type::const_pointer` `const_pointer`
- typedef `const value_type &` `const_reference`
- typedef `std::reverse_iterator< const_iterator >` `const_reverse_iterator`
- typedef `_CharT_alloc_type::difference_type` `difference_type`
- typedef `__gnu_cxx::__normal_iterator< pointer, __versa_string >` `iterator`
- typedef `_CharT_alloc_type::pointer` `pointer`
- typedef `value_type &` `reference`
- typedef `std::reverse_iterator< iterator >` `reverse_iterator`
- typedef `_CharT_alloc_type::size_type` `size_type`
- typedef `_Traits` `traits_type`
- typedef `_Traits::char_type` `value_type`

Public Member Functions

- `__versa_string` (const `_Alloc` & `a`=`_Alloc`()) noexcept
- `__versa_string` (const `__versa_string` & `str`)
- `__versa_string` (`__versa_string` && `str`) noexcept
- `__versa_string` (std::initializer_list< `_CharT` > `l`, const `_Alloc` & `a`=`_Alloc`())
- `__versa_string` (const `__versa_string` & `str`, size_type `pos`, size_type `n`=npos)
- `__versa_string` (const `__versa_string` & `str`, size_type `pos`, size_type `n`, const `_Alloc` & `a`)
- `__versa_string` (const `_CharT` * `s`, size_type `n`, const `_Alloc` & `a`=`_Alloc`())
- `__versa_string` (const `_CharT` * `s`, const `_Alloc` & `a`=`_Alloc`())
- `__versa_string` (size_type `n`, `_CharT` `c`, const `_Alloc` & `a`=`_Alloc`())
- template<class `_InputIterator` , typename = std::RequireInputIter< `_InputIterator` >>
`__versa_string` (`_InputIterator` `beg`, `_InputIterator` `end`, const `_Alloc` & `a`=`_Alloc`())
- `~__versa_string` () noexcept
- template<typename `_InputIterator` >
`__versa_string`< `_CharT`, `_Traits`, `_Alloc`, `_Base` > & **`M_replace_dispatch`** (const_iterator `i1`, const_iterator `i2`, `_InputIterator` `k1`, `_InputIterator` `k2`, std::false_type)
- `__versa_string` & `append` (const `__versa_string` & `str`)
- `__versa_string` & `append` (const `__versa_string` & `str`, size_type `pos`, size_type `n`)
- `__versa_string` & `append` (const `_CharT` * `s`, size_type `n`)
- `__versa_string` & `append` (const `_CharT` * `s`)
- `__versa_string` & `append` (size_type `n`, `_CharT` `c`)
- `__versa_string` & `append` (std::initializer_list< `_CharT` > `l`)
- template<class `_InputIterator` , typename = std::RequireInputIter< `_InputIterator` >>
`__versa_string` & `append` (`_InputIterator` `first`, `_InputIterator` `last`)
- `__versa_string` & `assign` (const `__versa_string` & `str`)
- `__versa_string` & `assign` (`__versa_string` && `str`) noexcept
- `__versa_string` & `assign` (const `__versa_string` & `str`, size_type `pos`, size_type `n`)
- `__versa_string` & `assign` (const `_CharT` * `s`, size_type `n`)
- `__versa_string` & `assign` (const `_CharT` * `s`)
- `__versa_string` & `assign` (size_type `n`, `_CharT` `c`)
- template<class `_InputIterator` , typename = std::RequireInputIter< `_InputIterator` >>
`__versa_string` & `assign` (`_InputIterator` `first`, `_InputIterator` `last`)
- `__versa_string` & `assign` (std::initializer_list< `_CharT` > `l`)
- const_reference `at` (size_type `n`) const
- reference `at` (size_type `n`)
- reference `back` () noexcept
- const_reference `back` () const noexcept
- iterator `begin` () noexcept
- const_iterator `begin` () const noexcept
- const `_CharT` * `c_str` () const noexcept
- size_type `capacity` () const noexcept
- const_iterator `cbegin` () const noexcept
- const_iterator `cend` () const noexcept
- void `clear` () noexcept
- int `compare` (const `__versa_string` & `str`) const
- int `compare` (size_type `pos`, size_type `n`, const `__versa_string` & `str`) const
- int `compare` (size_type `pos1`, size_type `n1`, const `__versa_string` & `str`, size_type `pos2`, size_type `n2`) const
- int `compare` (const `_CharT` * `s`) const
- int `compare` (size_type `pos`, size_type `n1`, const `_CharT` * `s`) const

- int [compare](#) (size_type __pos, size_type __n1, const _CharT *__s, size_type __n2) const
- size_type [copy](#) (_CharT *__s, size_type __n, size_type __pos=0) const
- [const_reverse_iterator crbegin](#) () const noexcept
- [const_reverse_iterator crend](#) () const noexcept
- const _CharT * [data](#) () const noexcept
- bool [empty](#) () const noexcept
- iterator [end](#) () noexcept
- const_iterator [end](#) () const noexcept
- [__versa_string & erase](#) (size_type __pos=0, size_type __n=npos)
- iterator [erase](#) (const_iterator __position)
- iterator [erase](#) (const_iterator __first, const_iterator __last)
- size_type [find](#) (const _CharT *__s, size_type __pos, size_type __n) const
- size_type [find](#) (const [__versa_string](#) & __str, size_type __pos=0) const noexcept
- size_type [find](#) (const _CharT *__s, size_type __pos=0) const
- size_type [find](#) (_CharT __c, size_type __pos=0) const noexcept
- size_type [find_first_not_of](#) (const [__versa_string](#) & __str, size_type __pos=0) const noexcept
- size_type [find_first_not_of](#) (const _CharT *__s, size_type __pos, size_type __n) const
- size_type [find_first_not_of](#) (const _CharT *__s, size_type __pos=0) const
- size_type [find_first_not_of](#) (_CharT __c, size_type __pos=0) const noexcept
- size_type [find_first_of](#) (const [__versa_string](#) & __str, size_type __pos=0) const noexcept
- size_type [find_first_of](#) (const _CharT *__s, size_type __pos, size_type __n) const
- size_type [find_first_of](#) (const _CharT *__s, size_type __pos=0) const
- size_type [find_first_of](#) (_CharT __c, size_type __pos=0) const noexcept
- size_type [find_last_not_of](#) (const [__versa_string](#) & __str, size_type __pos=npos) const noexcept
- size_type [find_last_not_of](#) (const _CharT *__s, size_type __pos, size_type __n) const
- size_type [find_last_not_of](#) (const _CharT *__s, size_type __pos=npos) const
- size_type [find_last_not_of](#) (_CharT __c, size_type __pos=npos) const noexcept
- size_type [find_last_of](#) (const [__versa_string](#) & __str, size_type __pos=npos) const noexcept
- size_type [find_last_of](#) (const _CharT *__s, size_type __pos, size_type __n) const
- size_type [find_last_of](#) (const _CharT *__s, size_type __pos=npos) const
- size_type [find_last_of](#) (_CharT __c, size_type __pos=npos) const noexcept
- reference [front](#) () noexcept
- const_reference [front](#) () const noexcept
- allocator_type [get_allocator](#) () const noexcept
- iterator [insert](#) (const_iterator __p, size_type __n, _CharT __c)
- template<class _InputIterator, typename = std::::RequireInputIter<_InputIterator>>>
iterator [insert](#) (const_iterator __p, _InputIterator __beg, _InputIterator __end)
- iterator [insert](#) (const_iterator __p, [std::initializer_list](#)<_CharT> __l)
- [__versa_string & insert](#) (size_type __pos1, const [__versa_string](#) & __str)
- [__versa_string & insert](#) (size_type __pos1, const [__versa_string](#) & __str, size_type __pos2, size_type __n)
- [__versa_string & insert](#) (size_type __pos, const _CharT *__s, size_type __n)
- [__versa_string & insert](#) (size_type __pos, const _CharT *__s)
- [__versa_string & insert](#) (size_type __pos, size_type __n, _CharT __c)
- iterator [insert](#) (const_iterator __p, _CharT __c)
- size_type [length](#) () const noexcept
- size_type [max_size](#) () const noexcept
- [__versa_string & operator+=](#) (const [__versa_string](#) & __str)
- [__versa_string & operator+=](#) (const _CharT *__s)
- [__versa_string & operator+=](#) (_CharT __c)
- [__versa_string & operator+=](#) ([std::initializer_list](#)<_CharT> __l)
- [__versa_string & operator=](#) (const [__versa_string](#) & __str)

- `__versa_string & operator= (__versa_string &&__str)` noexcept
- `__versa_string & operator= (std::initializer_list<_CharT > __l)`
- `__versa_string & operator= (const _CharT *__s)`
- `__versa_string & operator= (_CharT __c)`
- `const_reference operator[] (size_type __pos)` const noexcept
- `reference operator[] (size_type __pos)` noexcept
- `void pop_back ()`
- `void push_back (_CharT __c)`
- `reverse_iterator rbegin ()` noexcept
- `const_reverse_iterator rbegin ()` const noexcept
- `reverse_iterator rend ()` noexcept
- `const_reverse_iterator rend ()` const noexcept
- `__versa_string & replace (size_type __pos, size_type __n, const __versa_string &__str)`
- `__versa_string & replace (size_type __pos1, size_type __n1, const __versa_string &__str, size_type __pos2, size_type __n2)`
- `__versa_string & replace (size_type __pos, size_type __n1, const _CharT *__s, size_type __n2)`
- `__versa_string & replace (size_type __pos, size_type __n1, const _CharT *__s)`
- `__versa_string & replace (size_type __pos, size_type __n1, size_type __n2, _CharT __c)`
- `__versa_string & replace (const_iterator __i1, const_iterator __i2, const __versa_string &__str)`
- `__versa_string & replace (const_iterator __i1, const_iterator __i2, const _CharT *__s, size_type __n)`
- `__versa_string & replace (const_iterator __i1, const_iterator __i2, const _CharT *__s)`
- `__versa_string & replace (const_iterator __i1, const_iterator __i2, size_type __n, _CharT __c)`
- `template<class _InputIterator, typename = std::_RequireInputIter<_InputIterator>>>`
`__versa_string & replace (const_iterator __i1, const_iterator __i2, _InputIterator __k1, _InputIterator __k2)`
- `__versa_string & replace (const_iterator __i1, const_iterator __i2, _CharT *__k1, _CharT *__k2)`
- `__versa_string & replace (const_iterator __i1, const_iterator __i2, const _CharT *__k1, const _CharT *__k2)`
- `__versa_string & replace (const_iterator __i1, const_iterator __i2, iterator __k1, iterator __k2)`
- `__versa_string & replace (const_iterator __i1, const_iterator __i2, const_iterator __k1, const_iterator __k2)`
- `__versa_string & replace (const_iterator __i1, const_iterator __i2, std::initializer_list<_CharT > __l)`
- `void reserve (size_type __res_arg=0)`
- `void resize (size_type __n, _CharT __c)`
- `void resize (size_type __n)`
- `size_type rfind (const __versa_string &__str, size_type __pos=npow)` const noexcept
- `size_type rfind (const _CharT *__s, size_type __pos, size_type __n)` const
- `size_type rfind (const _CharT *__s, size_type __pos=npow)` const
- `size_type rfind (_CharT __c, size_type __pos=npow)` const noexcept
- `void shrink_to_fit ()` noexcept
- `size_type size ()` const noexcept
- `__versa_string substr (size_type __pos=0, size_type __n=npow)` const
- `void swap (__versa_string &__s)` noexcept

Static Public Attributes

- static const size_type `npos`

5.18.1 Detailed Description

```
template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>
class __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >
```

Template class `__versa_string`.

Data structure managing sequences of characters and character-like objects.

Definition at line 56 of file `vstring.h`.

5.18.2 Constructor & Destructor Documentation

```
5.18.2.1 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::__versa_string ( const _Alloc & __a =
_Alloc() ) [inline], [explicit], [noexcept]
```

Construct an empty string using allocator `a`.

Definition at line 137 of file `vstring.h`.

Referenced by `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::substr()`.

```
5.18.2.2 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename >
class _Base> __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::__versa_string ( const
__versa_string< _CharT, _Traits, _Alloc, _Base > & __str ) [inline]
```

Construct string with copy of value of `__str`.

Parameters

<code>__str</code>	Source string.
--------------------	----------------

Definition at line 145 of file `vstring.h`.

```
5.18.2.3 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::__versa_string ( __versa_string<
_CharT, _Traits, _Alloc, _Base > && __str ) [inline], [noexcept]
```

String move constructor.

Parameters

<code>__str</code>	Source string.
--------------------	----------------

The newly-constructed string contains the exact contents of `__str`. The contents of `__str` are a valid, but unspecified

string.

Definition at line 157 of file `vstring.h`.

```
5.18.2.4 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::__versa_string ( std::initializer_list<
_CharT> __l, const _Alloc & __a = _Alloc() ) [inline]
```

Construct string from an initializer list.

Parameters

<code>__l</code>	std::initializer_list of characters.
<code>__a</code>	Allocator to use (default is default allocator).

Definition at line 165 of file `vstring.h`.

```
5.18.2.5 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename >
class _Base> __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::__versa_string ( const
__versa_string<_CharT, _Traits, _Alloc, _Base> & __str, size_type __pos, size_type __n = npos ) [inline]
```

Construct string as copy of a substring.

Parameters

<code>__str</code>	Source string.
<code>__pos</code>	Index of first character to copy from.
<code>__n</code>	Number of characters to copy (default remainder).

Definition at line 176 of file `vstring.h`.

```
5.18.2.6 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename >
class _Base> __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::__versa_string ( const
__versa_string<_CharT, _Traits, _Alloc, _Base> & __str, size_type __pos, size_type __n, const _Alloc & __a )
[inline]
```

Construct string as copy of a substring.

Parameters

<code>__str</code>	Source string.
<code>__pos</code>	Index of first character to copy from.
<code>__n</code>	Number of characters to copy.
<code>__a</code>	Allocator to use.

Definition at line 191 of file vstring.h.

```
5.18.2.7  template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
           _Base> __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::__versa_string ( const _CharT * __s,
           size_type __n, const _Alloc & __a = _Alloc() ) [inline]
```

Construct string initialized by a character array.

Parameters

<code>__s</code>	Source character array.
<code>__n</code>	Number of characters to copy.
<code>__a</code>	Allocator to use (default is default allocator).

NB: `__s` must have at least `__n` characters, `'\0'` has no special meaning.

Definition at line 208 of file vstring.h.

```
5.18.2.8  template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
           _Base> __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::__versa_string ( const _CharT * __s,
           const _Alloc & __a = _Alloc() ) [inline]
```

Construct string as copy of a C string.

Parameters

<code>__s</code>	Source C string.
<code>__a</code>	Allocator to use (default is default allocator).

Definition at line 217 of file vstring.h.

```
5.18.2.9  template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
           _Base> __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::__versa_string ( size_type __n, _CharT
           __c, const _Alloc & __a = _Alloc() ) [inline]
```

Construct string as multiple characters.

Parameters

<code>__n</code>	Number of characters.
<code>__c</code>	Character to use.
<code>__a</code>	Allocator to use (default is default allocator).

Definition at line 227 of file `vstring.h`.

```
5.18.2.10 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename
> class _Base> template<class _InputIterator, typename = std::_RequireInputIter<_InputIterator>>
__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::__versa_string ( _InputIterator __beg,
_InputIterator __end, const _Alloc & __a = _Alloc() ) [inline]
```

Construct string as copy of a range.

Parameters

<code>__beg</code>	Start of range.
<code>__end</code>	End of range.
<code>__a</code>	Allocator to use (default is default allocator).

Definition at line 242 of file `vstring.h`.

```
5.18.2.11 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::~~__versa_string ( ) [inline],
[noexcept]
```

Destroy the string instance.

Definition at line 249 of file `vstring.h`.

5.18.3 Member Function Documentation

```
5.18.3.1 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::append ( const
__versa_string<_CharT, _Traits, _Alloc, _Base> & __str ) [inline]
```

Append a string to this string.

Parameters

<code>__str</code>	The string to append.
--------------------	-----------------------

Returns

Reference to this string.

Definition at line 692 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

Referenced by `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::append()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare()`, `std::getline()`, `__gnu_cxx::operator+()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator+=()`.

5.18.3.2 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::append (const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str, size_type __pos, size_type __n) [inline]`

Append a substring.

Parameters

<code>__str</code>	The string to append.
<code>__pos</code>	Index of the first character of <code>str</code> to append.
<code>__n</code>	The number of characters to append.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	if <code>pos</code> is not a valid index.
--------------------------------	---

This function appends `__n` characters from `__str` starting at `__pos` to this string. If `__n` is larger than the number of available characters in `__str`, the remainder of `__str` is appended.

Definition at line 709 of file `vstring.h`.

5.18.3.3 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::append (const _CharT* __s, size_type __n) [inline]`

Append a C substring.

Parameters

<code>__s</code>	The C string to append.
<code>__n</code>	The number of characters to append.

Returns

Reference to this string.

Definition at line 721 of file `vstring.h`.

5.18.3.4 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::append (const _CharT * __s) [inline]`

Append a C string.

Parameters

<code>__s</code>	The C string to append.
------------------	-------------------------

Returns

Reference to this string.

Definition at line 734 of file `vstring.h`.

```
5.18.3.5 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> __versa_string& __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::append ( size_type __n,
_CharT __c ) [inline]
```

Append multiple characters.

Parameters

<code>__n</code>	The number of characters to append.
<code>__c</code>	The character to use.

Returns

Reference to this string.

Appends `n` copies of `c` to this string.

Definition at line 751 of file `vstring.h`.

References `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::size()`.

```
5.18.3.6 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename >
class _Base> __versa_string& __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::append (
std::initializer_list<_CharT> __l ) [inline]
```

Append an `initializer_list` of characters.

Parameters

<code>l</code>	The <code>initializer_list</code> of characters to append.
----------------	--

Returns

Reference to this string.

Definition at line 761 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::append()`.

```
5.18.3.7 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
        _Base> template<class _InputIterator, typename = std:: RequireInputIter<_InputIterator>> __versa_string&
        __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::append ( _InputIterator __first, _InputIterator __last )
        [inline]
```

Append a range of characters.

Parameters

<code>__first</code>	Iterator referencing the first character to append.
<code>__last</code>	Iterator marking the end of the range.

Returns

Reference to this string.

Appends characters in the range `[first,last)` to this string.

Definition at line 780 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::replace()`.

```
5.18.3.8 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
        _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::assign ( const
        __versa_string<_CharT, _Traits, _Alloc, _Base > & __str ) [inline]
```

Set value to contents of another string.

Parameters

<code>__str</code>	Source string to use.
--------------------	-----------------------

Returns

Reference to this string.

Definition at line 803 of file `vstring.h`.

Referenced by `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::assign()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::operator=()`.

5.18.3.9 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::assign (__versa_string< _CharT, _Traits, _Alloc, _Base > && __str) [inline], [noexcept]`

Set value to contents of another string.

Parameters

<code>__str</code>	Source string to use.
--------------------	-----------------------

Returns

Reference to this string.

This function sets this string to the exact contents of `__str`. `__str` is a valid, but unspecified string.

Definition at line 819 of file `vstring.h`.

References `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::swap()`.

5.18.3.10 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::assign (const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str, size_type __pos, size_type __n) [inline]`

Set value to a substring of a string.

Parameters

<code>__str</code>	The string to use.
<code>__pos</code>	Index of the first character of str.
<code>__n</code>	Number of characters to use.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	if <code>__pos</code> is not a valid index.
--------------------------------	---

This function sets this string to the substring of `__str` consisting of `__n` characters at `__pos`. If `__n` is larger than the number of available characters in `__str`, the remainder of `__str` is used.

Definition at line 840 of file `vstring.h`.

References `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::size()`.

```
5.18.3.11 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::assign ( const _CharT
*_s, size_type __n ) [inline]
```

Set value to a C substring.

Parameters

<code>__s</code>	The C string to use.
<code>__n</code>	Number of characters to use.

Returns

Reference to this string.

This function sets the value of this string to the first `__n` characters of `__s`. If `__n` is larger than the number of available characters in `__s`, the remainder of `__s` is used.

Definition at line 857 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::size()`.

```
5.18.3.12 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::assign ( const _CharT
*_s ) [inline]
```

Set value to contents of a C string.

Parameters

<code>__s</code>	The C string to use.
------------------	----------------------

Returns

Reference to this string.

This function sets the value of this string to the value of `__s`. The data is copied, so there is no dependence on `__s` once the function returns.

Definition at line 873 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::size()`.

```
5.18.3.13 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::assign ( size_type __n,
_CharT __c ) [inline]
```

Set value to multiple characters.

Parameters

<code>__n</code>	Length of the resulting string.
<code>__c</code>	The character to use.

Returns

Reference to this string.

This function sets the value of this string to `__n` copies of character `__c`.

Definition at line 890 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::size()`.

```
5.18.3.14 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> template<class _InputIterator, typename = std::RequireInputIter<_InputIterator>> __versa_string&
__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::assign ( _InputIterator __first, _InputIterator __last )
[inline]
```

Set value to a range of characters.

Parameters

<code>__first</code>	Iterator referencing the first character to append.
<code>__last</code>	Iterator marking the end of the range.

Returns

Reference to this string.

Sets value of string to characters in the range `[first,last)`.

Definition at line 909 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::replace()`.

```
5.18.3.15 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename >
class _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::assign (
std::initializer_list<_CharT> __l ) [inline]
```

Set value to an `initializer_list` of characters.

Parameters

<code>↔</code>	The initializer_list of characters to assign.
<code>↔</code>	
<code>↔</code>	
<code>↔</code>	
<code>/</code>	

Returns

Reference to this string.

Definition at line 919 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::assign()`.

5.18.3.16 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> const_reference __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::at (size_type __n) const [inline]`

Provides access to the data contained in the string.

Parameters

<code>↔</code>	The index of the character to access.
<code>__n</code>	

Returns

Read-only (const) reference to the character.

Exceptions

<code>std::out_of_range</code>	If <code>__n</code> is an invalid index.
--------------------------------	--

This function provides for safer data access. The parameter is first checked that it is in the range of the string. The function throws `out_of_range` if the check fails.

Definition at line 577 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

5.18.3.17 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> reference __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::at (size_type __n) [inline]`

Provides access to the data contained in the string.

Parameters

<code>_↔</code>	The index of the character to access.
<code>_n</code>	

Returns

Read/write reference to the character.

Exceptions

<code>std::out_of_range</code>	If <code>__n</code> is an invalid index.
--------------------------------	--

This function provides for safer data access. The parameter is first checked that it is in the range of the string. The function throws `out_of_range` if the check fails. Success results in unsharing the string.

Definition at line 599 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

5.18.3.18 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> reference __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::back () [inline], [noexcept]`

Returns a read/write reference to the data at the last element of the string.

Definition at line 632 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator[]()`, and `__gnu_cxx::__versa_↔string<_CharT, _Traits, _Alloc, _Base>::size()`.

5.18.3.19 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> const_reference __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::back () const [inline], [noexcept]`

Returns a read-only (constant) reference to the data at the last element of the string.

Definition at line 640 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator[]()`, and `__gnu_cxx::__versa_↔string<_CharT, _Traits, _Alloc, _Base>::size()`.

5.18.3.20 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> iterator __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::begin () [inline], [noexcept]`

Returns a read/write iterator that points to the first character in the string. Unshares the string.

Definition at line 315 of file `vstring.h`.

Referenced by `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::crend()`, and `__gnu_cxx::__versa_↔string<_CharT, _Traits, _Alloc, _Base>::rend()`.

```
5.18.3.21 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename >
class _Base> const_iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::begin ( ) const
[inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first character in the string.

Definition at line 326 of file `vstring.h`.

```
5.18.3.22 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> const _CharT* __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::c_str ( ) const
[inline], [noexcept]
```

Return const pointer to null-terminated contents.

This is a handle to internal data. Do not modify or dire things may happen.

Definition at line 1647 of file `vstring.h`.

```
5.18.3.23 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename >
class _Base> size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::capacity ( ) const
[inline], [noexcept]
```

Returns the total number of characters that the string can hold before needing to allocate more memory.

Definition at line 486 of file `vstring.h`.

Referenced by `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::push_back()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::shrink_to_fit()`.

```
5.18.3.24 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> const_iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::cbegin ( ) const
[inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first character in the string.

Definition at line 390 of file `vstring.h`.

```
5.18.3.25 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename >
class _Base> const_iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::cend ( ) const
[inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last character in the string.

Definition at line 398 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

```
5.18.3.26 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename >
class _Base> void __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::clear ( ) [inline],
[noexcept]
```

Erases the string, making it empty.

Definition at line 514 of file `vstring.h`.

```
5.18.3.27 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> int __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::compare ( const __versa_string<
_CharT, _Traits, _Alloc, _Base > & __str ) const [inline]
```

Compare to a string.

Parameters

<code>__str</code>	String to compare against.
--------------------	----------------------------

Returns

Integer < 0, 0, or > 0.

Returns an integer < 0 if this string is ordered before `__str`, 0 if their values are equivalent, or > 0 if this string is ordered after `__str`. Determines the effective length `rlen` of the strings to compare as the smallest of `size()` and `str.size()`. The function then compares the two strings by calling `traits::compare(data(), str.data(), rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

Definition at line 2073 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::append()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::capacity()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::data()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::insert()`, `std::min()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

Referenced by `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_last_not_of()`, `__gnu_cxx::operator<()`, `__gnu_cxx::operator<=()`, `__gnu_cxx::operator==()`, `__gnu_cxx::operator>()`, and `__gnu_cxx::operator>=()`.

5.18.3.28 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> int __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare (size_type __pos, size_type __n, const __versa_string<_CharT, _Traits, _Alloc, _Base> & __str) const`

Compare substring to a string.

Parameters

<code>__pos</code>	Index of first character of substring.
<code>__n</code>	Number of characters in substring.
<code>__str</code>	String to compare against.

Returns

Integer < 0, 0, or > 0.

Form the substring of this string from the `__n` characters starting at `__pos`. Returns an integer < 0 if the substring is ordered before `__str`, 0 if their values are equivalent, or > 0 if the substring is ordered after `__str`. Determines the effective length `rlen` of the strings to compare as the smallest of the length of the substring and `__str.size()`. The function then compares the two strings by calling `traits::compare(substring.data(), str.data(), rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

Definition at line 460 of file `vstring.tcc`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::data()`, `std::min()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

5.18.3.29 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> int __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::compare (size_type __pos1, size_type __n1, const __versa_string<_CharT, _Traits, _Alloc, _Base > & __str, size_type __pos2, size_type __n2) const`

Compare substring to a substring.

Parameters

<code>__pos1</code>	Index of first character of substring.
<code>__n1</code>	Number of characters in substring.
<code>__str</code>	String to compare against.
<code>__pos2</code>	Index of first character of substring of str.
<code>__n2</code>	Number of characters in substring of str.

Returns

Integer < 0, 0, or > 0.

Form the substring of this string from the `__n1` characters starting at `__pos1`. Form the substring of `__str` from the `__n2` characters starting at `__pos2`. Returns an integer < 0 if this substring is ordered before the substring of `__str`, 0 if their values are equivalent, or > 0 if this substring is ordered after the substring of `__str`. Determines the effective length `rlen` of the strings to compare as the smallest of the lengths of the substrings. The function then compares the two strings by calling `traits::compare(substring.data(), str.substr(pos2, n2).data(), rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

Definition at line 477 of file `vstring.tcc`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::compare()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::data()`, and `std::min()`.

5.18.3.30 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> int __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::compare (const _CharT * __s) const`

Compare to a C string.

Parameters

<code>__s</code>	C string to compare against.
------------------	------------------------------

Returns

Integer < 0, 0, or > 0.

Returns an integer < 0 if this string is ordered before `__s`, 0 if their values are equivalent, or > 0 if this string is ordered after `__s`. Determines the effective length `rlen` of the strings to compare as the smallest of `size()` and the length of a string constructed from `__s`. The function then compares the two strings by calling `traits::compare(data(), s, rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

Definition at line 496 of file vstring.tcc.

References `std::min()`.

5.18.3.31 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> int __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::compare (size_type __pos, size_type __n1, const _CharT * __s) const`

Compare substring to a C string.

Parameters

<code>__pos</code>	Index of first character of substring.
<code>__n1</code>	Number of characters in substring.
<code>__s</code>	C string to compare against.

Returns

Integer < 0 , 0 , or > 0 .

Form the substring of this string from the `__n1` characters starting at `__pos`. Returns an integer < 0 if the substring is ordered before `__s`, 0 if their values are equivalent, or > 0 if the substring is ordered after `__s`. Determines the effective length `rlen` of the strings to compare as the smallest of the length of the substring and the length of a string constructed from `__s`. The function then compares the two string by calling `traits::compare(substring.data(),s,rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

Definition at line 512 of file vstring.tcc.

References `std::min()`.

5.18.3.32 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> int __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::compare (size_type __pos, size_type __n1, const _CharT * __s, size_type __n2) const`

Compare substring against a character array.

Parameters

<code>__pos</code>	Index of first character of substring.
<code>__n1</code>	Number of characters in substring.
<code>__s</code>	character array to compare against.
<code>__n2</code>	Number of characters of <code>s</code> .

Returns

Integer < 0 , 0 , or > 0 .

Form the substring of this string from the `__n1` characters starting at `__pos`. Form a string from the first `__n2` characters of `__s`. Returns an integer < 0 if this substring is ordered before the string from `__s`, 0 if their values are equivalent, or

> 0 if this substring is ordered after the string from `__s`. Determines the effective length `rlen` of the strings to compare as the smallest of the length of the substring and `__n2`. The function then compares the two strings by calling `traits::compare(substring.data(), __s, rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

NB: `__s` must have at least `n2` characters, `l0` has no special meaning.

Definition at line 529 of file `vstring.tcc`.

References `std::min()`.

5.18.3.33 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string<_CharT, _Traits, _Alloc, _Base>::size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::copy (_CharT * __s, size_type __n, size_type __pos = 0) const`

Copy substring into C string.

Parameters

<code>__s</code>	C string to copy value into.
<code>__n</code>	Number of characters to copy.
<code>__pos</code>	Index of first character to copy.

Returns

Number of characters actually copied

Exceptions

<code>std::out_of_range</code>	If <code>pos > size()</code> .
--------------------------------	-----------------------------------

Copies up to `__n` characters starting at `__pos` into the C string `s`. If `__pos` is greater than `size()`, `out_of_range` is thrown.

Definition at line 255 of file `vstring.tcc`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find()`.

Referenced by `__gnu_cxx::operator+()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace()`.

5.18.3.34 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> const_reverse_iterator __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::rbegin () const [inline], [noexcept]`

Returns a read-only (constant) reverse iterator that points to the last character in the string. Iteration is done in reverse element order.

Definition at line 407 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::end()`.

5.18.3.35 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> const_reverse_iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::crend () const [inline], [noexcept]`

Returns a read-only (constant) reverse iterator that points to one before the first character in the string. Iteration is done in reverse element order.

Definition at line 416 of file `vstring.h`.

References `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::begin()`.

5.18.3.36 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> const _CharT* __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::data () const [inline], [noexcept]`

Return const pointer to contents.

This is a handle to internal data. Do not modify or dire things may happen.

Definition at line 1657 of file `vstring.h`.

Referenced by `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::compare()`, `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find()`, `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_first_not_of()`, `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_first_of()`, `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_last_not_of()`, `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_last_of()`, `std::operator<()`, and `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::rfind()`.

5.18.3.37 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> bool __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::empty () const [inline], [noexcept]`

Returns true if the string is empty. Equivalent to `*this == ""`.

Definition at line 522 of file `vstring.h`.

References `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::size()`.

5.18.3.38 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::end () [inline], [noexcept]`

Returns a read/write iterator that points one past the last character in the string. Unshares the string.

Definition at line 334 of file `vstring.h`.

References `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::size()`.

Referenced by `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::cbegin()`, and `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::rbegin()`.

5.18.3.39 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> const_iterator __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::end () const`
`[inline], [noexcept]`

Returns a read-only (constant) iterator that points one past the last character in the string.

Definition at line 345 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::size()`.

5.18.3.40 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::erase (size_type __pos = 0, size_type __n = npos)` `[inline]`

Remove characters.

Parameters

<code>__pos</code>	Index of first character to remove (default 0).
<code>__n</code>	Number of characters to remove (default remainder).

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <code>__pos</code> is beyond the end of this string.
--------------------------------	---

Removes `__n` characters from this string starting at `__pos`. The length of the string is reduced by `__n`. If there are `< __n` characters to remove, the remainder of the string is truncated. If `__p` is beyond end of string, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 1173 of file `vstring.h`.

Referenced by `std::getline()`.

5.18.3.41 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> iterator __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::erase (const_iterator __position)`
`[inline]`

Remove one character.

Parameters

<code>__position</code>	Iterator referencing the character to remove.
-------------------------	---

Returns

iterator referencing same location after removal.

Removes the character at `__position` from this string. The value of the string doesn't change if an error is thrown.

Definition at line 1190 of file `vstring.h`.

```
5.18.3.42  template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
           _Base> iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::erase ( const_iterator __first,
           const_iterator __last ) [inline]
```

Remove a range of characters.

Parameters

<code>__first</code>	Iterator referencing the first character to remove.
<code>__last</code>	Iterator referencing the end of the range.

Returns

Iterator referencing location of first after removal.

Removes the characters in the range `[first,last)` from this string. The value of the string doesn't change if an error is thrown.

Definition at line 1215 of file `vstring.h`.

```
5.18.3.43  template<typename _CharT, typename _Traits , typename _Alloc , template< typename, typename, typename > class
           _Base> __versa_string< _CharT, _Traits, _Alloc, _Base >::size_type __gnu_cxx::__versa_string< _CharT,
           _Traits, _Alloc, _Base >::find ( const _CharT * __s, size_type __pos, size_type __n ) const
```

Find position of a C substring.

Parameters

<code>__s</code>	C string to locate.
<code>__pos</code>	Index of character to search from.
<code>__n</code>	Number of characters from <code>__s</code> to search for.

Returns

Index of start of first occurrence.

Starting from `__pos`, searches forward for the first `__n` characters in `__s` within this string. If found, returns the index where it begins. If not found, returns `npos`.

Definition at line 270 of file `vstring.tcc`.

Referenced by `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::copy()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_first_of()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::get_allocator()`.

5.18.3.44 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find (const __versa_string<_CharT, _Traits, _Alloc, _Base> & __str, size_type __pos = 0) const` `[inline]`, `[noexcept]`

Find position of a string.

Parameters

<code>__str</code>	String to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of start of first occurrence.

Starting from `__pos`, searches forward for value of `__str` within this string. If found, returns the index where it begins. If not found, returns `npos`.

Definition at line 1693 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::data()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

5.18.3.45 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find (const _CharT * __s, size_type __pos = 0) const` `[inline]`

Find position of a C string.

Parameters

<code>__s</code>	C string to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of start of first occurrence.

Starting from `__pos`, searches forward for the value of `__s` within this string. If found, returns the index where it begins. If not found, returns `npos`.

Definition at line 1708 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find()`.

5.18.3.46 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string< _CharT, _Traits, _Alloc, _Base >::size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find (_CharT __c, size_type __pos = 0) const [noexcept]`

Find position of a character.

Parameters

<code>__c</code>	Character to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 294 of file `vstring.tcc`.

References `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::rfind()`.

5.18.3.47 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_first_not_of (const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str, size_type __pos = 0) const [inline], [noexcept]`

Find position of a character not in string.

Parameters

<code>__str</code>	String containing characters to avoid.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for a character not contained in `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 1925 of file `vstring.h`.

References `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::data()`, and `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::size()`.

Referenced by `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::find_first_not_of()`, and `__gnu_cxx::__↵_versa_string< _CharT, _Traits, _Alloc, _Base >::find_last_of()`.

5.18.3.48 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string<_CharT, _Traits, _Alloc, _Base>::size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_first_not_of(const _CharT* __s, size_type __pos, size_type __n) const`

Find position of a character not in C substring.

Parameters

<code>__s</code>	C string containing characters to avoid.
<code>__pos</code>	Index of character to search from.
<code>__n</code>	Number of characters from <code>s</code> to consider.

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for a character not contained in the first `__n` characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 392 of file `vstring.tcc`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_first_not_of()`.

5.18.3.49 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_first_not_of(const _CharT* __s, size_type __pos = 0) const [inline]`

Find position of a character not in C string.

Parameters

<code>__s</code>	C string containing characters to avoid.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for a character not contained in `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 1956 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_first_not_of()`.

5.18.3.50 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string<_CharT, _Traits, _Alloc, _Base>::size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_first_not_of(_CharT __c, size_type __pos = 0) const [noexcept]`

Find position of a different character.

Parameters

<code>__c</code>	Character to avoid.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for a character other than `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 405 of file `vstring.tcc`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_last_not_of()`.

```
5.18.3.51  template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename
> class _Base> size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_first_of (
const __versa_string<_CharT, _Traits, _Alloc, _Base> & __str, size_type __pos = 0 ) const    [inline],
[noexcept]
```

Find position of a character of string.

Parameters

<code>__str</code>	String containing characters to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for one of the characters of `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 1798 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::data()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

Referenced by `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_first_of()`, and `__gnu_cxx::__↵
versa_string<_CharT, _Traits, _Alloc, _Base>::rfind()`.

```
5.18.3.52  template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> __versa_string<_CharT, _Traits, _Alloc, _Base>::size_type __gnu_cxx::__versa_string<_CharT,
_Traits, _Alloc, _Base>::find_first_of ( const _CharT * __s, size_type __pos, size_type __n ) const
```

Find position of a character of C substring.

Parameters

<code>__s</code>	String containing characters to locate.
<code>__pos</code>	Index of character to search from.
<code>__n</code>	Number of characters from <code>s</code> to search for.

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for one of the first `__n` characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 353 of file `vstring.tcc`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::find_last_of()`.

5.18.3.53 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::find_first_of (const _CharT * __s, size_type __pos = 0) const [inline]`

Find position of a character of C string.

Parameters

<code>__s</code>	String containing characters to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for one of the characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 1828 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::find_first_of()`.

5.18.3.54 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::find_first_of (_CharT __c, size_type __pos = 0) const [inline], [noexcept]`

Find position of a character.

Parameters

<code>__c</code>	Character to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for the character `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Note: equivalent to `find(c, pos)`.

Definition at line 1847 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find()`.

```
5.18.3.55 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
    _Base> size_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base>::find_last_not_of( const
    __versa_string< _CharT, _Traits, _Alloc, _Base> & __str, size_type __pos = npos ) const    [inline],
    [noexcept]
```

Find last position of a character not in string.

Parameters

<code>__str</code>	String containing characters to avoid.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for a character not contained in `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 1988 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::data()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

Referenced by `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_first_not_of()`, and `__gnu_cxx::__↵_versa_string<_CharT, _Traits, _Alloc, _Base>::find_last_not_of()`.

```
5.18.3.56 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
    _Base> __versa_string< _CharT, _Traits, _Alloc, _Base>::size_type __gnu_cxx::__versa_string< _CharT,
    _Traits, _Alloc, _Base>::find_last_not_of( const _CharT * __s, size_type __pos, size_type __n ) const
```

Find last position of a character not in C substring.

Parameters

<code>__s</code>	C string containing characters to avoid.
<code>__pos</code>	Index of character to search back from.
<code>__n</code>	Number of characters from <code>s</code> to consider.

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for a character not contained in the first `__n` characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 417 of file `vstring.tcc`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_last_not_of()`.

```
5.18.3.57 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_last_not_of ( const _CharT *
__s, size_type __pos = npos ) const [inline]
```

Find last position of a character not in C string.

Parameters

<code>__s</code>	C string containing characters to avoid.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for a character not contained in `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 2019 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_last_not_of()`.

```
5.18.3.58 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> __versa_string<_CharT, _Traits, _Alloc, _Base>::size_type __gnu_cxx::__versa_string<_CharT,
_Traits, _Alloc, _Base>::find_last_not_of ( _CharT __c, size_type __pos = npos ) const [noexcept]
```

Find last position of a different character.

Parameters

<code>__c</code>	Character to avoid.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for a character other than `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 439 of file `vstring.tcc`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare()`.

```
5.18.3.59 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename >
class _Base> size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_last_of( const
__versa_string<_CharT, _Traits, _Alloc, _Base> & __str, size_type __pos = npos ) const [inline],
[noexcept]
```

Find last position of a character of string.

Parameters

<code>__str</code>	String containing characters to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for one of the characters of `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 1862 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::data()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

Referenced by `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_first_of()`, and `__gnu_cxx::__↵
versa_string<_CharT, _Traits, _Alloc, _Base>::find_last_of()`.

```
5.18.3.60 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> __versa_string<_CharT, _Traits, _Alloc, _Base>::size_type __gnu_cxx::__versa_string<_CharT,
_Traits, _Alloc, _Base>::find_last_of( const _CharT * __s, size_type __pos, size_type __n ) const
```

Find last position of a character of C substring.

Parameters

<code>__s</code>	C string containing characters to locate.
<code>__pos</code>	Index of character to search back from.
<code>__n</code>	Number of characters from <code>s</code> to search for.

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for one of the first `__n` characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 370 of file `vstring.tcc`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_first_not_of()`.

```
5.18.3.61  template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
           _Base> size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_last_of ( const _CharT * __s,
           size_type __pos = npos ) const    [inline]
```

Find last position of a character of C string.

Parameters

<code>__s</code>	C string containing characters to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for one of the characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 1892 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_last_of()`.

```
5.18.3.62  template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
           _Base> size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_last_of ( _CharT __c,
           size_type __pos = npos ) const    [inline], [noexcept]
```

Find last position of a character.

Parameters

<code>__c</code>	Character to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Note: equivalent to `rfind(c, pos)`.

Definition at line 1911 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::rfind()`.

5.18.3.63 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> reference __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::front () [inline], [noexcept]`

Returns a read/write reference to the data at the first element of the string.

Definition at line 616 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator[]()`.

5.18.3.64 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> const_reference __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::front () const [inline], [noexcept]`

Returns a read-only (constant) reference to the data at the first element of the string.

Definition at line 624 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator[]()`.

5.18.3.65 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> allocator_type __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::get_allocator () const [inline], [noexcept]`

Return copy of allocator used to construct this string.

Definition at line 1664 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find()`.

5.18.3.66 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::insert (const_iterator __p, size_type __n, _CharT __c) [inline]`

Insert multiple characters.

Parameters

<code>__p</code>	Const_iterator referencing location in string to insert at.
<code>__n</code>	Number of characters to insert
<code>__c</code>	The character to insert.

Returns

Iterator referencing the first inserted char.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts `__n` copies of character `__c` starting at the position referenced by iterator `__p`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 940 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::replace()`.

Referenced by `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::compare()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::insert()`.

5.18.3.67 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> template<class _InputIterator, typename = std::RequireInputIter<_InputIterator>> iterator __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::insert (const_iterator __p, _InputIterator __beg, _InputIterator __end) [inline]`

Insert a range of characters.

Parameters

<code>__p</code>	Const_iterator referencing location in string to insert at.
<code>__beg</code>	Start of range.
<code>__end</code>	End of range.

Returns

Iterator referencing the first inserted char.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts characters in range `[beg,end)`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 984 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::insert()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::replace()`.

5.18.3.68 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::insert (const_iterator __p, std::initializer_list< _CharT > __l) [inline]`

Insert an initializer_list of characters.

Parameters

<code>__p</code>	Const_iterator referencing location in string to insert at.
<code>__l</code>	The initializer_list of characters to insert.

Returns

Iterator referencing the first inserted char.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Definition at line 1020 of file `vstring.h`.

References `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::insert()`.

5.18.3.69 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::insert (size_type __pos1, const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str) [inline]`

Insert value of a string.

Parameters

<code>__pos1</code>	Iterator referencing location in string to insert at.
<code>__str</code>	The string to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts value of `__str` starting at `__pos1`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 1037 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

5.18.3.70 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::insert (size_type __pos1, const __versa_string<_CharT, _Traits, _Alloc, _Base> &__str, size_type __pos2, size_type __n) [inline]`

Insert a substring.

Parameters

<code>__pos1</code>	Iterator referencing location in string to insert at.
<code>__str</code>	The string to insert.
<code>__pos2</code>	Start of characters in <code>str</code> to insert.
<code>__n</code>	Number of characters to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
<code>std::out_of_range</code>	If <code>__pos1 > size()</code> or <code>__pos2 > __str.size()</code> .

Starting at `__pos1`, insert `__n` character of `__str` beginning with `__pos2`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. If `__pos1` is beyond the end of this string or `__pos2` is beyond the end of `__str`, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 1060 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace()`.

5.18.3.71 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::insert (size_type __pos, const _CharT* __s, size_type __n) [inline]`

Insert a C substring.

Parameters

<code>__pos</code>	Iterator referencing location in string to insert at.
<code>__s</code>	The C string to insert.
<code>__n</code>	The number of characters to insert.

Returns

Reference to this string.

Exceptions

<i>std::length_error</i>	If new length exceeds <code>max_size()</code> .
<i>std::out_of_range</i>	If <code>__pos</code> is beyond the end of this string.

Inserts the first `__n` characters of `__s` starting at `__pos`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. If `__pos` is beyond `end()`, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 1083 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace()`.

5.18.3.72 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::insert (size_type __pos, const _CharT* __s) [inline]`

Insert a C string.

Parameters

<code>__pos</code>	Iterator referencing location in string to insert at.
<code>__s</code>	The C string to insert.

Returns

Reference to this string.

Exceptions

<i>std::length_error</i>	If new length exceeds <code>max_size()</code> .
<i>std::out_of_range</i>	If <code>__pos</code> is beyond the end of this string.

Inserts the first `__n` characters of `__s` starting at `__pos`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. If `__pos` is beyond `end()`, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 1102 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace()`.

5.18.3.73 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::insert (size_type __pos, size_type __n, _CharT __c) [inline]`

Insert multiple characters.

Parameters

<code>__pos</code>	Index in string to insert at.
<code>__n</code>	Number of characters to insert
<code>__c</code>	The character to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
<code>std::out_of_range</code>	If <code>__pos</code> is beyond the end of this string.

Inserts `__n` copies of character `__c` starting at index `__pos`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. If `__pos > length()`, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 1126 of file `vstring.h`.

```
5.18.3.74 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
    _Base> iterator __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::insert ( const_iterator __p, _CharT
    __c ) [inline]
```

Insert one character.

Parameters

<code>__p</code>	Iterator referencing position in string to insert at.
<code>__c</code>	The character to insert.

Returns

Iterator referencing newly inserted char.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts character `__c` at position referenced by `__p`. If adding character causes the length to exceed `max_size()`, `length_error` is thrown. If `__p` is beyond end of string, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 1145 of file `vstring.h`.

5.18.3.75 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::length () const [inline], [noexcept]`

Returns the number of characters in the string, not including any null-termination.

Definition at line 431 of file `vstring.h`.

5.18.3.76 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::max_size () const [inline], [noexcept]`

Returns the `size()` of the largest possible string.

Definition at line 436 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::resize()`.

Referenced by `std::getline()`.

5.18.3.77 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator+=(const __versa_string<_CharT, _Traits, _Alloc, _Base> &__str) [inline]`

Append a string to this string.

Parameters

<code>__str</code>	The string to append.
--------------------	-----------------------

Returns

Reference to this string.

Definition at line 651 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::append()`.

5.18.3.78 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator+=(const _CharT *__s) [inline]`

Append a C string.

Parameters

<code>__↵ __s</code>	The C string to append.
--------------------------	-------------------------

Returns

Reference to this string.

Definition at line 660 of file vstring.h.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::append()`.

5.18.3.79 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::operator+=(_CharT _c) [inline]`

Append a character.

Parameters

<code>_c</code>	The character to append.
-----------------	--------------------------

Returns

Reference to this string.

Definition at line 669 of file vstring.h.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::push_back()`.

5.18.3.80 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::operator+=(std::initializer_list<_CharT> _l) [inline]`

Append an `initializer_list` of characters.

Parameters

<code>_l</code>	The <code>initializer_list</code> of characters to be appended.
-----------------	---

Returns

Reference to this string.

Definition at line 682 of file vstring.h.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::append()`.

5.18.3.81 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator= (const __versa_string<_CharT, _Traits, _Alloc, _Base> &__str) [inline]`

Assign the value of *str* to this string.

Parameters

<code>__str</code>	Source string.
--------------------	----------------

Definition at line 256 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::assign()`.

5.18.3.82 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator= (__versa_string<_CharT, _Traits, _Alloc, _Base> &&__str) [inline], [noexcept]`

String move assignment operator.

Parameters

<code>__str</code>	Source string.
--------------------	----------------

The contents of `__str` are moved into this string (without copying). `__str` is a valid, but unspecified string.

Definition at line 268 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::swap()`.

5.18.3.83 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator= (std::initializer_list<_CharT> __l) [inline]`

Set value to string constructed from initializer list.

Parameters

<code>↵</code>	<code>std::initializer_list.</code>
<code>__↵</code>	
<code>↵</code>	
<code>__↵</code>	
<code>/</code>	

Definition at line 280 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::assign()`.

5.18.3.84 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::operator= (const _CharT * __s) [inline]`

Copy contents of `__s` into this string.

Parameters

<code>__s</code>	Source null-terminated string.
------------------	--------------------------------

Definition at line 292 of file `vstring.h`.

References `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::assign()`.

5.18.3.85 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::operator= (_CharT __c) [inline]`

Set value to string of length 1.

Parameters

<code>__c</code>	Source character.
------------------	-------------------

Assigning to a character makes this string length 1 and `(*this)[0] == __c`.

Definition at line 303 of file `vstring.h`.

References `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::assign()`.

5.18.3.86 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> const_reference __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::operator[] (size_type __pos) const [inline], [noexcept]`

Subscript access to the data contained in the string.

Parameters

<code>__pos</code>	The index of the character to access.
--------------------	---------------------------------------

Returns

Read-only (constant) reference to the character.

This operator allows for easy, array-style, data access. Note that data access with this operator is unchecked and `out_of_range` lookups are not defined. (For checked lookups see `at()`.)

Definition at line 537 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

Referenced by `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::back()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::front()`.

5.18.3.87 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> reference __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator[] (size_type __pos) [inline], [noexcept]`

Subscript access to the data contained in the string.

Parameters

<code>__pos</code>	The index of the character to access.
--------------------	---------------------------------------

Returns

Read/write reference to the character.

This operator allows for easy, array-style, data access. Note that data access with this operator is unchecked and `out_of_range` lookups are not defined. (For checked lookups see `at()`.) Unshares the string.

Definition at line 554 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

5.18.3.88 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> void __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::pop_back () [inline]`

Remove the last character.

The string must be non-empty.

Definition at line 1235 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

5.18.3.89 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> void __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::push_back (_CharT __c) [inline]`

Append a single character.

Parameters

<code>__c</code>	Character to append.
------------------	----------------------

Definition at line 788 of file vstring.h.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::capacity()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

Referenced by `__gnu_cxx::operator+()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator+=()`.

5.18.3.90 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> reverse_iterator __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::rbegin ()`
`[inline], [noexcept]`

Returns a read/write reverse iterator that points to the last character in the string. Iteration is done in reverse element order. Unshares the string.

Definition at line 354 of file vstring.h.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::end()`.

5.18.3.91 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> const_reverse_iterator __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::rbegin ()`
`const [inline], [noexcept]`

Returns a read-only (constant) reverse iterator that points to the last character in the string. Iteration is done in reverse element order.

Definition at line 363 of file vstring.h.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::end()`.

5.18.3.92 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> reverse_iterator __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::rend ()`
`[inline], [noexcept]`

Returns a read/write reverse iterator that points to one before the first character in the string. Iteration is done in reverse element order. Unshares the string.

Definition at line 372 of file vstring.h.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::begin()`.

5.18.3.93 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> const_reverse_iterator __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::rend () const`
`[inline], [noexcept]`

Returns a read-only (constant) reverse iterator that points to one before the first character in the string. Iteration is done in reverse element order.

Definition at line 381 of file vstring.h.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::begin()`.

5.18.3.94 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace (size_type _pos, size_type _n, const __versa_string<_CharT, _Traits, _Alloc, _Base> & _str) [inline]`

Replace characters with value from another string.

Parameters

<code>__pos</code>	Index of first character to replace.
<code>__n</code>	Number of characters to be replaced.
<code>__str</code>	String to insert.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <code>__pos</code> is beyond the end of this string.
<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range `[pos,pos+n)` from this string. In place, the value of `__str` is inserted. If `__pos` is beyond end of string, `out_of_range` is thrown. If the length of the result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 1257 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::size()`.

Referenced by `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::append()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::assign()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::insert()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::replace()`.

5.18.3.95 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::replace (size_type __pos1, size_type __n1, const __versa_string<_CharT, _Traits, _Alloc, _Base > & __str, size_type __pos2, size_type __n2) [inline]`

Replace characters with value from another string.

Parameters

<code>__pos1</code>	Index of first character to replace.
<code>__n1</code>	Number of characters to be replaced.
<code>__str</code>	String to insert.
<code>__pos2</code>	Index of first character of <code>str</code> to use.
<code>__n2</code>	Number of characters from <code>str</code> to use.

Returns

Reference to this string.

Exceptions

<i>std::out_of_range</i>	If <code>__pos1 > size()</code> or <code>__pos2 > str.size()</code> .
<i>std::length_error</i>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range `[pos1, pos1 + n)` from this string. In place, the value of `__str` is inserted. If `__pos` is beyond end of string, `out_of_range` is thrown. If the length of the result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 1280 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace()`.

5.18.3.96 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace (size_type __pos, size_type __n1, const _CharT * __s, size_type __n2) [inline]`

Replace characters with value of a C substring.

Parameters

<code>__pos</code>	Index of first character to replace.
<code>__n1</code>	Number of characters to be replaced.
<code>__s</code>	C string to insert.
<code>__n2</code>	Number of characters from <code>__s</code> to use.

Returns

Reference to this string.

Exceptions

<i>std::out_of_range</i>	If <code>__pos1 > size()</code> .
<i>std::length_error</i>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range `[pos, pos + n1)` from this string. In place, the first `__n2` characters of `__s` are inserted, or all of `__s` if `__n2` is too large. If `__pos` is beyond end of string, `out_of_range` is thrown. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 1308 of file `vstring.h`.

5.18.3.97 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace (size_type __pos, size_type __n1, const _CharT * __s) [inline]`

Replace characters with value of a C string.

Parameters

<code>__pos</code>	Index of first character to replace.
<code>__n1</code>	Number of characters to be replaced.
<code>__s</code>	C string to insert.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <code>__pos > size()</code> .
<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range `[pos, pos + n1)` from this string. In place, the characters of `__s` are inserted. If `pos` is beyond end of string, `out_of_range` is thrown. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 1332 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::replace()`.

5.18.3.98 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::replace (size_type __pos, size_type __n1, size_type __n2, _CharT __c) [inline]`

Replace characters with multiple characters.

Parameters

<code>__pos</code>	Index of first character to replace.
<code>__n1</code>	Number of characters to be replaced.
<code>__n2</code>	Number of characters to insert.
<code>__c</code>	Character to insert.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <code>__pos > size()</code> .
<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range [pos,pos + n1) from this string. In place, __n2 copies of __c are inserted. If __pos is beyond end of string, out_of_range is thrown. If the length of result exceeds max_size(), length_error is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 1356 of file vstring.h.

5.18.3.99 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::replace (const_iterator __i1, const_iterator __i2, const __versa_string< _CharT, _Traits, _Alloc, _Base > & __str) [inline]`

Replace range of characters with string.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__str</code>	String value to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range [i1,i2). In place, the value of __str is inserted. If the length of result exceeds max_size(), length_error is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 1375 of file vstring.h.

References `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::replace()`, and `__gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::size()`.

5.18.3.100 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::replace (const_iterator __i1, const_iterator __i2, const _CharT * __s, size_type __n) [inline]`

Replace range of characters with C substring.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__s</code>	C string value to insert.
<code>__n</code>	Number of characters from s to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[i1,i2)`. In place, the first *n* characters of `__s` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 1398 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::replace()`.

5.18.3.101 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::replace (const_iterator __i1, const_iterator __i2, const _CharT* __s) [inline]`

Replace range of characters with C string.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__s</code>	C string value to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[i1,i2)`. In place, the characters of `__s` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 1424 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::replace()`.

5.18.3.102 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename >
class _Base> __versa_string& __gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base >::replace (
const_iterator __i1, const_iterator __i2, size_type __n, _CharT __c) [inline]`

Replace range of characters with multiple characters.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__n</code>	Number of characters to insert.
<code>__c</code>	Character to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[i1,i2)`. In place, `__n` copies of `__c` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 1449 of file `vstring.h`.

```
5.18.3.103 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
    _Base> template<class _InputIterator, typename = std:: RequireInputIter<_InputIterator>>> __versa_string&
    __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::replace( const_iterator __i1, const_iterator __i2,
    _InputIterator __k1, _InputIterator __k2 ) [inline]
```

Replace range of characters with range.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__k1</code>	Iterator referencing start of range to insert.
<code>__k2</code>	Iterator referencing end of range to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range [i1,i2). In place, characters in the range [k1,k2) are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 1478 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::replace()`.

5.18.3.104 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string& __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::replace (const_iterator __i1, const_iterator __i2, std::initializer_list<_CharT > __l) [inline]`

Replace range of characters with `initializer_list`.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__l</code>	The <code>initializer_list</code> of characters to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range [i1,i2). In place, characters in the range [k1,k2) are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 1582 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::copy()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::replace()`.

5.18.3.105 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> void __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::reserve (size_type __res_arg = 0) [inline]`

Attempt to preallocate enough memory for specified number of characters.

Parameters

<code>__res_arg</code>	Number of characters required.
------------------------	--------------------------------

Exceptions

<code>std::length_error</code>	If <code>__res_arg</code> exceeds <code>max_size()</code> .
--------------------------------	---

This function attempts to reserve enough memory for the string to hold the specified number of characters. If the number requested is more than `max_size()`, `length_error` is thrown.

The advantage of this function is that if optimal code is a necessity and the user can determine the string length that will be required, the user can reserve the memory in advance, and thus prevent a possible reallocation of memory and copying of string data.

Definition at line 507 of file `vstring.h`.

Referenced by `__gnu_cxx::operator+()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::shrink_↵
to_fit()`.

5.18.3.106 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
_Base> void __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::resize (size_type __n, _CharT __c)`

Resizes the string to the specified number of characters.

Parameters

<code>↵ __n</code>	Number of characters the string should contain.
<code>↵ __c</code>	Character to fill any new elements.

This function will resize the string to the specified number of characters. If the number is smaller than the string's current size the string is truncated, otherwise the string is extended and new elements are set to `__c`.

Definition at line 50 of file `vstring.tcc`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::size()`.

Referenced by `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::max_size()`, and `__gnu_cxx::__versa↵
_string<_CharT, _Traits, _Alloc, _Base >::resize()`.

5.18.3.107 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename >
class _Base> void __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >::resize (size_type __n)
[inline]`

Resizes the string to the specified number of characters.

Parameters

<code>↵ __n</code>	Number of characters the string should contain.
------------------------	---

This function will resize the string to the specified length. If the new size is smaller than the string's current size the string is truncated, otherwise the string is extended and new characters are default-constructed. For basic types such as char, this means setting them to 0.

Definition at line 463 of file vstring.h.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::resize()`.

5.18.3.108 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::rfind (const __versa_string<_CharT, _Traits, _Alloc, _Base> & __str, size_type __pos = npos) const [inline], [noexcept]`

Find last position of a string.

Parameters

<code>__str</code>	String to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of start of last occurrence.

Starting from `__pos`, searches backward for value of `__str` within this string. If found, returns the index where it begins. If not found, returns `npos`.

Definition at line 1738 of file vstring.h.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::data()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

Referenced by `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_last_of()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::rfind()`.

5.18.3.109 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string<_CharT, _Traits, _Alloc, _Base>::size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::rfind (const _CharT* __s, size_type __pos, size_type __n) const`

Find last position of a C substring.

Parameters

<code>__s</code>	C string to locate.
<code>__pos</code>	Index of character to search back from.
<code>__n</code>	Number of characters from s to search for.

Returns

Index of start of last occurrence.

Starting from `__pos`, searches backward for the first `__n` characters in `__s` within this string. If found, returns the index where it begins. If not found, returns `npos`.

Definition at line 313 of file `vstring.tcc`.

References `std::min()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::rfind()`.

```
5.18.3.110 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
    _Base> size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::rfind ( const _CharT * __s,
    size_type __pos = npos ) const    [inline]
```

Find last position of a C string.

Parameters

<code>__s</code>	C string to locate.
<code>__pos</code>	Index of character to start search at (default end).

Returns

Index of start of last occurrence.

Starting from `__pos`, searches backward for the value of `__s` within this string. If found, returns the index where it begins. If not found, returns `npos`.

Definition at line 1768 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::rfind()`.

```
5.18.3.111 template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class
    _Base> __versa_string<_CharT, _Traits, _Alloc, _Base>::size_type __gnu_cxx::__versa_string<_CharT,
    _Traits, _Alloc, _Base>::rfind ( _CharT __c, size_type __pos = npos ) const    [noexcept]
```

Find last position of a character.

Parameters

<code>__c</code>	Character to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 335 of file `vstring.tcc`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_first_of()`.

5.18.3.112 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> void __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::shrink_to_fit () [inline], [noexcept]`

A non-binding request to reduce `capacity()` to `size()`.

Definition at line 469 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::capacity()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::reserve()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size()`.

5.18.3.113 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::size () const [inline], [noexcept]`

Returns the number of characters in the string, not including any null-termination.

Definition at line 425 of file `vstring.h`.

Referenced by `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::append()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::assign()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::at()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::back()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::cend()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::compare()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::empty()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::end()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_first_not_of()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_first_of()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_last_not_of()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::find_last_of()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::insert()`, `__gnu_cxx::operator+()`, `std::operator<<()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator[]()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::pop_back()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::push_back()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::replace()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::resize()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::rfind()`, and `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::shrink_to_fit()`.

5.18.3.114 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> __versa_string __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::substr (size_type __pos = 0, size_type __n = npos) const [inline]`

Get a substring.

Parameters

<code>__pos</code>	Index of first character (default 0).
<code>__n</code>	Number of characters in substring (default remainder).

Returns

The new string.

Exceptions

<code>std::out_of_range</code>	If <code>pos > size()</code> .
--------------------------------	-----------------------------------

Construct and return a new string using the `__n` characters starting at `__pos`. If the string is too short, use the remainder of the characters. If `__pos` is beyond the end of the string, `out_of_range` is thrown.

Definition at line 2052 of file `vstring.h`.

References `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::__versa_string()`.

5.18.3.115 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> void __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::swap (__versa_string<_CharT, _Traits, _Alloc, _Base> &__s) [inline], [noexcept]`

Swap contents with another string.

Parameters

<code>__s</code>	String to swap with.
------------------	----------------------

Exchanges the contents of this string with that of `__s` in constant time.

Definition at line 1636 of file `vstring.h`.

Referenced by `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::assign()`, `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::operator=()`, and `__gnu_cxx::swap()`.

5.18.4 Member Data Documentation

5.18.4.1 `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base> const __versa_string<_CharT, _Traits, _Alloc, _Base>::size_type __gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>::npos [static]`

Value returned by various member functions when they fail.

Definition at line 81 of file `vstring.h`.

The documentation for this class was generated from the following files:

- [vstring.h](#)
- [vstring.tcc](#)

5.19 `__gnu_cxx::_Caster<_ToType>` Struct Template Reference

Public Types

- `typedef _ToType::element_type * type`

5.19.1 Detailed Description

```
template<typename _ToType>
struct __gnu_cxx::_Caster<_ToType>
```

These functions are here to allow containers to support non standard pointer types. For normal pointers, these resolve to the use of the standard cast operation. For other types the functions will perform the appropriate cast to/from the custom pointer class so long as that class meets the following conditions: 1) has a typedef `element_type` which names the type it points to. 2) has a `get()` const method which returns `element_type*`. 3) has a constructor which can take one `element_type*` argument. This type supports the semantics of the pointer cast operators (below.)

Definition at line 52 of file `cast.h`.

The documentation for this struct was generated from the following file:

- [cast.h](#)

5.20 `__gnu_cxx::_Char_types<_CharT>` Struct Template Reference

Public Types

- `typedef unsigned long int_type`
- `typedef std::streamoff off_type`
- `typedef std::streampos pos_type`
- `typedef std::mbstate_t state_type`

5.20.1 Detailed Description

```
template<typename _CharT>
struct __gnu_cxx::_Char_types<_CharT>
```

Mapping from character type to associated types.

Note

This is an implementation class for the generic version of `char_traits`. It defines `int_type`, `off_type`, `pos_type`, and `state_type`. By default these are unsigned long, `streamoff`, `streampos`, and `mbstate_t`. Users who need a different set of types, but who don't need to change the definitions of any function defined in `char_traits`, can specialize `__gnu_cxx::_Char_types` while leaving `__gnu_cxx::char_traits` alone.

Definition at line 58 of file `char_traits.h`.

The documentation for this struct was generated from the following file:

- [char_traits.h](#)

5.21 `__gnu_cxx::_ExtPtr_allocator<_Tp>` Class Template Reference

Public Types

- typedef `_Pointer_adapter<_Relative_pointer_impl<const _Tp>>` `const_pointer`
- typedef `const _Tp &` `const_reference`
- typedef `std::ptrdiff_t` `difference_type`
- typedef `_Pointer_adapter<_Relative_pointer_impl<_Tp>>` `pointer`
- typedef `_Tp &` `reference`
- typedef `std::size_t` `size_type`
- typedef `_Tp` `value_type`

Public Member Functions

- `_ExtPtr_allocator` (`const _ExtPtr_allocator &__rarg`) `noexcept`
- `template<typename _Up>`
`_ExtPtr_allocator` (`const _ExtPtr_allocator<_Up> &__rarg`) `noexcept`
- `const std::allocator<_Tp> &_M_getUnderlyingImp ()` `const`
- `pointer address` (`reference __x`) `const noexcept`
- `const_pointer address` (`const_reference __x`) `const noexcept`
- `pointer allocate` (`size_type __n`, `void *__hint=0`)
- `template<typename _Up, typename... _Args>`
`void construct` (`_Up *__p`, `_Args &&... __args`)
- `template<typename... _Args>`
`void construct` (`pointer __p`, `_Args &&... __args`)
- `void deallocate` (`pointer __p`, `size_type __n`)
- `template<typename _Up>`
`void destroy` (`_Up *__p`)
- `void destroy` (`pointer __p`)
- `size_type max_size ()` `const noexcept`
- `template<typename _Up>`
`bool operator!=` (`const _ExtPtr_allocator<_Up> &__rarg`)
- `bool operator!=` (`const _ExtPtr_allocator &__rarg`)
- `template<typename _Up>`
`bool operator==` (`const _ExtPtr_allocator<_Up> &__rarg`)
- `bool operator==` (`const _ExtPtr_allocator &__rarg`)

Friends

- `template<typename _Up>`
`void swap` (`_ExtPtr_allocator<_Up> &`, `_ExtPtr_allocator<_Up> &`)

5.21.1 Detailed Description

```
template<typename _Tp>
class __gnu_cxx::_ExtPtr_allocator< _Tp >
```

An example allocator which uses a non-standard pointer type.

This allocator specifies that containers use a 'relative pointer' as it's pointer type. (See `ext/pointer.h`) Memory allocation in this example is still performed using `std::allocator`.

Definition at line 56 of file `extptr_allocator.h`.

The documentation for this class was generated from the following file:

- [extptr_allocator.h](#)

5.22 __gnu_cxx::_Invalid_type Struct Reference

5.22.1 Detailed Description

The specialization on this type helps resolve the problem of reference to void, and eliminates the need to specialize `_Pointer_adapter` for cases of `void*`, `const void*`, and so on.

Definition at line 213 of file `pointer.h`.

The documentation for this struct was generated from the following file:

- [pointer.h](#)

5.23 __gnu_cxx::_Pointer_adapter< _Storage_policy > Class Template Reference

Inherits `_Storage_policy`.

Public Types

- typedef `std::ptrdiff_t` **difference_type**
- typedef `_Storage_policy::element_type` **element_type**
- typedef [std::random_access_iterator_tag](#) **iterator_category**
- typedef `_Pointer_adapter` **pointer**
- typedef `_Reference_type< element_type >::reference` **reference**
- typedef [_Unqualified_type](#)< `element_type` >::type **value_type**

Public Member Functions

- `_Pointer_adapter` (element_type * __arg=0)
- `_Pointer_adapter` (const `_Pointer_adapter` & __arg)
- `template<typename _Up>`
`_Pointer_adapter` (_Up * __arg)
- `template<typename _Up>`
`_Pointer_adapter` (const `_Pointer_adapter`<_Up> & __arg)
- `operator __unspecified_bool_type` () const
- `bool operator!` () const
- `reference operator*` () const
- `_Pointer_adapter` & `operator++` ()
- `_Pointer_adapter` `operator++` (int)
- `_Pointer_adapter` & `operator+=` (short __offset)
- `_Pointer_adapter` & `operator+=` (unsigned short __offset)
- `_Pointer_adapter` & `operator+=` (int __offset)
- `_Pointer_adapter` & `operator+=` (unsigned int __offset)
- `_Pointer_adapter` & `operator+=` (long __offset)
- `_Pointer_adapter` & `operator+=` (unsigned long __offset)
- `template<typename _Up>`
`std::ptrdiff_t operator-` (const `_Pointer_adapter`<_Up> & __rhs) const
- `_Pointer_adapter` & `operator--` ()
- `_Pointer_adapter` `operator--` (int)
- `_Pointer_adapter` & `operator-=` (short __offset)
- `_Pointer_adapter` & `operator-=` (unsigned short __offset)
- `_Pointer_adapter` & `operator-=` (int __offset)
- `_Pointer_adapter` & `operator-=` (unsigned int __offset)
- `_Pointer_adapter` & `operator-=` (long __offset)
- `_Pointer_adapter` & `operator-=` (unsigned long __offset)
- `element_type * operator->` () const
- `_Pointer_adapter` & `operator=` (const `_Pointer_adapter` & __arg)
- `template<typename _Up>`
`_Pointer_adapter` & `operator=` (const `_Pointer_adapter`<_Up> & __arg)
- `template<typename _Up>`
`_Pointer_adapter` & `operator=` (_Up * __arg)
- `reference operator[]` (std::ptrdiff_t __index) const

Friends

- `_Pointer_adapter` `operator+` (const `_Pointer_adapter` & __lhs, short __offset)
- `_Pointer_adapter` `operator+` (short __offset, const `_Pointer_adapter` & __rhs)
- `_Pointer_adapter` `operator+` (const `_Pointer_adapter` & __lhs, unsigned short __offset)
- `_Pointer_adapter` `operator+` (unsigned short __offset, const `_Pointer_adapter` & __rhs)
- `_Pointer_adapter` `operator+` (const `_Pointer_adapter` & __lhs, int __offset)
- `_Pointer_adapter` `operator+` (int __offset, const `_Pointer_adapter` & __rhs)
- `_Pointer_adapter` `operator+` (const `_Pointer_adapter` & __lhs, unsigned int __offset)
- `_Pointer_adapter` `operator+` (unsigned int __offset, const `_Pointer_adapter` & __rhs)
- `_Pointer_adapter` `operator+` (const `_Pointer_adapter` & __lhs, long __offset)
- `_Pointer_adapter` `operator+` (long __offset, const `_Pointer_adapter` & __rhs)
- `_Pointer_adapter` `operator+` (unsigned long __offset, const `_Pointer_adapter` & __rhs)

- [_Pointer_adapter](#) **operator+** (const [_Pointer_adapter](#) &__lhs, unsigned long __offset)
- `std::ptrdiff_t` **operator-** (const [_Pointer_adapter](#) &__lhs, `element_type` *__rhs)
- `std::ptrdiff_t` **operator-** (`element_type` *__lhs, const [_Pointer_adapter](#) &__rhs)
- `template<typename _Up >`
`std::ptrdiff_t` **operator-** (const [_Pointer_adapter](#) &__lhs, `_Up` *__rhs)
- `template<typename _Up >`
`std::ptrdiff_t` **operator-** (`_Up` *__lhs, const [_Pointer_adapter](#) &__rhs)
- [_Pointer_adapter](#) **operator** (const [_Pointer_adapter](#) &__lhs, short __offset)
- [_Pointer_adapter](#) **operator-** (const [_Pointer_adapter](#) &__lhs, unsigned short __offset)
- [_Pointer_adapter](#) **operator-** (const [_Pointer_adapter](#) &__lhs, int __offset)
- [_Pointer_adapter](#) **operator-** (const [_Pointer_adapter](#) &__lhs, unsigned int __offset)
- [_Pointer_adapter](#) **operator-** (const [_Pointer_adapter](#) &__lhs, long __offset)
- [_Pointer_adapter](#) **operator-** (const [_Pointer_adapter](#) &__lhs, unsigned long __offset)

5.23.1 Detailed Description

```
template<typename _Storage_policy>
class __gnu_cxx::_Pointer_adapter<_Storage_policy >
```

The following provides an 'alternative pointer' that works with the containers when specified as the pointer typedef of the allocator.

The pointer type used with the containers doesn't have to be this class, but it must support the implicit conversions, pointer arithmetic, comparison operators, etc. that are supported by this class, and avoid raising compile-time ambiguities. Because creating a working pointer can be challenging, this pointer template was designed to wrapper an easier storage policy type, so that it becomes reusable for creating other pointer types.

A key point of this class is also that it allows container writers to 'assume' `Allocator::pointer` is a typedef for a normal pointer. This class supports most of the conventions of a true pointer, and can, for instance handle implicit conversion to const and base class pointer types. The only impositions on container writers to support extended pointers are: 1) use the `Allocator::pointer` typedef appropriately for pointer types. 2) if you need pointer casting, use the `__pointer_cast<>` functions from `ext/cast.h`. This allows pointer cast operations to be overloaded as necessary by custom pointers.

Note: The const qualifier works with this pointer adapter as follows:

```
_Tp* == _Pointer_adapter<_Std_pointer_impl<_Tp> >; const _Tp* == _Pointer_adapter<_Std_pointer_impl<const
_Tp> >; _Tp* const == const _Pointer_adapter<_Std_pointer_impl<_Tp> >; const _Tp* const == const _Pointer_
adapter<_Std_pointer_impl<const _Tp> >;
```

Definition at line 281 of file `pointer.h`.

The documentation for this class was generated from the following file:

- [pointer.h](#)

5.24 `__gnu_cxx::Relative_pointer_impl<_Tp>` Class Template Reference

Public Types

- typedef `_Tp` **element_type**

Public Member Functions

- `_Tp * get () const`
- `bool operator< (const _Relative_pointer_impl &__rarg) const`
- `bool operator== (const _Relative_pointer_impl &__rarg) const`
- `void set (_Tp *__arg)`

5.24.1 Detailed Description

```
template<typename _Tp>
class __gnu_cxx::Relative_pointer_impl< _Tp >
```

A storage policy for use with `_Pointer_adapter<>` which stores the pointer's address as an offset value which is relative to its own address.

This is intended for pointers within shared memory regions which might be mapped at different addresses by different processes. For null pointers, a value of 1 is used. (0 is legitimate sometimes for nodes in circularly linked lists) This value was chosen as the least likely to generate an incorrect null, As there is no reason why any normal pointer would point 1 byte into its own pointer address.

Definition at line 109 of file `pointer.h`.

The documentation for this class was generated from the following file:

- [pointer.h](#)

5.25 `__gnu_cxx::Relative_pointer_impl< const _Tp >` Class Template Reference

Public Types

- `typedef const _Tp element_type`

Public Member Functions

- `const _Tp * get () const`
- `bool operator< (const _Relative_pointer_impl &__rarg) const`
- `bool operator== (const _Relative_pointer_impl &__rarg) const`
- `void set (const _Tp *__arg)`

5.25.1 Detailed Description

```
template<typename _Tp>
class __gnu_cxx::Relative_pointer_impl< const _Tp >
```

`Relative_pointer_impl` needs a specialization for `const T` because of the casting done during pointer arithmetic.

Definition at line 161 of file `pointer.h`.

The documentation for this class was generated from the following file:

- [pointer.h](#)

5.26 `__gnu_cxx::Std_pointer_impl<_Tp>` Class Template Reference

Public Types

- typedef `_Tp` **element_type**

Public Member Functions

- `_Tp * get () const`
- bool **operator**< (const `_Std_pointer_impl` &__rarg) const
- bool **operator**== (const `_Std_pointer_impl` &__rarg) const
- void **set** (element_type *__arg)

5.26.1 Detailed Description

```
template<typename _Tp>
class __gnu_cxx::Std_pointer_impl<_Tp>
```

A storage policy for use with `_Pointer_adapter<>` which yields a standard pointer.

A `_Storage_policy` is required to provide 4 things: 1) A `get()` API for returning the stored pointer value. 2) An `set()` API for storing a pointer value. 3) An `element_type` typedef to define the type this points to. 4) An `operator<()` to support pointer comparison. 5) An `operator==()` to support pointer comparison.

Definition at line 66 of file `pointer.h`.

The documentation for this class was generated from the following file:

- [pointer.h](#)

5.27 `__gnu_cxx::Unqualified_type<_Tp>` Struct Template Reference

Public Types

- typedef `_Tp` **type**

5.27.1 Detailed Description

```
template<typename _Tp>
struct __gnu_cxx::Unqualified_type<_Tp>
```

This structure accommodates the way in which `std::iterator_traits<>` is normally specialized for `const T*`, so that `value_of<_type>` is still `T`.

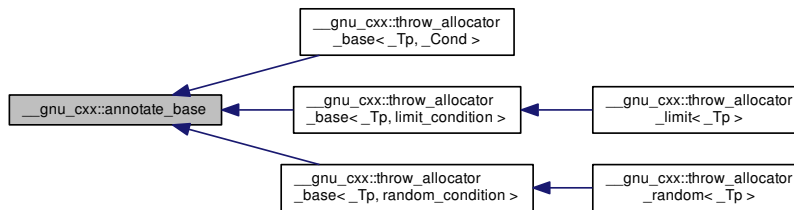
Definition at line 241 of file `pointer.h`.

The documentation for this struct was generated from the following file:

- [pointer.h](#)

5.28 `__gnu_cxx::annotate_base` Struct Reference

Inheritance diagram for `__gnu_cxx::annotate_base`:



Public Member Functions

- void **check** (size_t label)
- void **check_allocated** (void *p, size_t size)
- void **check_constructed** (void *p)
- void **check_constructed** (size_t label)
- void **erase** (void *p, size_t size)
- void **erase_construct** (void *p)
- void **insert** (void *p, size_t size)
- void **insert_construct** (void *p)

Static Public Member Functions

- static void **check** ()
- static size_t **get_label** ()
- static void **set_label** (size_t l)

Friends

- `std::ostream` & **operator**<< (`std::ostream` &, const `annotate_base` &)

5.28.1 Detailed Description

Base class for checking address and label information about allocations. Create a `std::map` between the allocated address (void*) and a datum for annotations, which are a pair of numbers corresponding to label and allocated size.

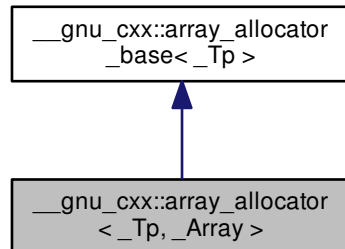
Definition at line 88 of file `throw_allocator.h`.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

5.29 `__gnu_cxx::array_allocator<_Tp, _Array>` Class Template Reference

Inheritance diagram for `__gnu_cxx::array_allocator<_Tp, _Array>`:



Public Types

- typedef `_Array` **array_type**
- typedef const `_Tp *` **const_pointer**
- typedef const `_Tp &` **const_reference**
- typedef `ptrdiff_t` **difference_type**
- typedef `std::true_type` **is_always_equal**
- typedef `_Tp *` **pointer**
- typedef `std::true_type` **propagate_on_container_move_assignment**
- typedef `_Tp &` **reference**
- typedef `size_t` **size_type**
- typedef `_Tp` **value_type**

Public Member Functions

- **array_allocator** (`array_type *` __array=0) noexcept
- **array_allocator** (const `array_allocator` &__o) noexcept
- template<typename `_Tp1` , typename `_Array1` >
array_allocator (const `array_allocator`< `_Tp1`, `_Array1` > &) noexcept
- `pointer` **address** (`reference` __x) const noexcept
- `const_pointer` **address** (`const_reference` __x) const noexcept
- `pointer` **allocate** (`size_type` __n, const void *=0)
- template<typename `_Up` , typename... `_Args`>
void **construct** (`_Up *` __p, `_Args` &&... __args)
- void **deallocate** (`pointer`, `size_type`)
- template<typename `_Up` >
void **destroy** (`_Up *` __p)
- `size_type` **max_size** () const noexcept

5.29.1 Detailed Description

```
template<typename _Tp, typename _Array = std::tr1::array<_Tp, 1>>
class __gnu_cxx::array_allocator<_Tp, _Array>
```

An allocator that uses previously allocated memory. This memory can be externally, globally, or otherwise allocated.

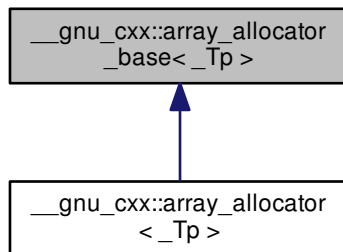
Definition at line 110 of file `array_allocator.h`.

The documentation for this class was generated from the following file:

- [array_allocator.h](#)

5.30 `__gnu_cxx::array_allocator_base<_Tp>` Class Template Reference

Inheritance diagram for `__gnu_cxx::array_allocator_base<_Tp>`:



Public Types

- `typedef const _Tp * const_pointer`
- `typedef const _Tp & const_reference`
- `typedef ptrdiff_t difference_type`
- `typedef _Tp * pointer`
- `typedef _Tp & reference`
- `typedef size_t size_type`
- `typedef _Tp value_type`

Public Member Functions

- pointer **address** (reference __x) const noexcept
- const_pointer **address** (const_reference __x) const noexcept
- template<typename _Up, typename... _Args>
void **construct** (_Up *__p, _Args &&... __args)
- void **deallocate** (pointer, size_type)
- template<typename _Up >
void **destroy** (_Up *__p)
- size_type **max_size** () const noexcept

5.30.1 Detailed Description

```
template<typename _Tp>
class __gnu_cxx::array_allocator_base< _Tp >
```

Base class.

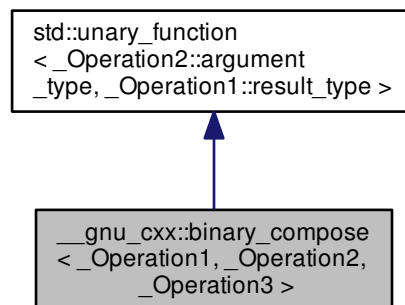
Definition at line 54 of file array_allocator.h.

The documentation for this class was generated from the following file:

- [array_allocator.h](#)

5.31 __gnu_cxx::binary_compose< _Operation1, _Operation2, _Operation3 > Class Template Reference

Inheritance diagram for __gnu_cxx::binary_compose< _Operation1, _Operation2, _Operation3 >:



Public Types

- typedef `_Arg` [argument_type](#)
- typedef `_Result` [result_type](#)

Public Member Functions

- **`binary_compose`** (const `_Operation1` &__x, const `_Operation2` &__y, const `_Operation3` &__z)
- `_Operation1::result_type` **`operator()`** (const typename `_Operation2::argument_type` &__x) const

Protected Attributes

- `_Operation1` **`_M_fn1`**
- `_Operation2` **`_M_fn2`**
- `_Operation3` **`_M_fn3`**

5.31.1 Detailed Description

```
template<class _Operation1, class _Operation2, class _Operation3>
class __gnu_cxx::binary_compose<_Operation1, _Operation2, _Operation3 >
```

An [SGI extension](#) .

Definition at line 150 of file `ext/functional`.

5.31.2 Member Typedef Documentation

5.31.2.1 `template<typename _Arg, typename _Result> typedef _Arg std::unary_function<_Arg, _Result >::argument_type`
[[inherited](#)]

`argument_type` is the type of the argument

Definition at line 108 of file `stl_function.h`.

5.31.2.2 `template<typename _Arg, typename _Result> typedef _Result std::unary_function<_Arg, _Result >::result_type`
[[inherited](#)]

`result_type` is the return type

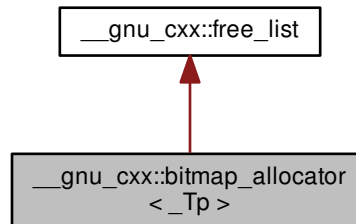
Definition at line 111 of file `stl_function.h`.

The documentation for this class was generated from the following file:

- [ext/functional](#)

5.32 `__gnu_cxx::bitmap_allocator<_Tp>` Class Template Reference

Inheritance diagram for `__gnu_cxx::bitmap_allocator<_Tp>`:



Public Types

- typedef `free_list::__mutex_type` **__mutex_type**
- typedef `const _Tp *` **const_pointer**
- typedef `const _Tp &` **const_reference**
- typedef `ptrdiff_t` **difference_type**
- typedef `_Tp *` **pointer**
- typedef `std::true_type` **propagate_on_container_move_assignment**
- typedef `_Tp &` **reference**
- typedef `size_t` **size_type**
- typedef `_Tp` **value_type**

Public Member Functions

- **bitmap_allocator** (const `bitmap_allocator` &) noexcept
- template<typename `_Tp1` >
 bitmap_allocator (const `bitmap_allocator`< `_Tp1` > &) noexcept
- pointer `_M_allocate_single_object` () throw (std::bad_alloc)
- void `_M_deallocate_single_object` (pointer `__p`) throw ()
- pointer **address** (reference `__r`) const noexcept
- const_pointer **address** (const_reference `__r`) const noexcept
- pointer **allocate** (size_type `__n`)
- pointer **allocate** (size_type `__n`, typename `bitmap_allocator`< void >::const_pointer)
- template<typename `_Up`, typename... `_Args`>
 void **construct** (`_Up` *`__p`, `_Args` &&... `__args`)
- void **deallocate** (pointer `__p`, size_type `__n`) throw ()
- template<typename `_Up` >
 void **destroy** (`_Up` *`__p`)
- size_type **max_size** () const noexcept

Private Types

- typedef `vector_type::iterator` **iterator**
- typedef `__detail::__mini_vector<value_type>` **vector_type**

Private Member Functions

- void `_M_clear()`
- `size_t * _M_get(size_t __sz) throw (std::bad_alloc)`
- void `_M_insert(size_t * __addr) throw ()`

5.32.1 Detailed Description

```
template<typename _Tp>
class __gnu_cxx::bitmap_allocator<_Tp>
```

Bitmap Allocator, primary template.

Definition at line 663 of file `bitmap_allocator.h`.

5.32.2 Member Function Documentation

5.32.2.1 `template<typename _Tp> pointer __gnu_cxx::bitmap_allocator<_Tp>::_M_allocate_single_object () throw std::bad_alloc) [inline]`

Allocates memory for a single object of size `sizeof(_Tp)`.

Exceptions

<code>std::bad_alloc.</code>	If memory can not be allocated.
------------------------------	---------------------------------

Complexity: Worst case complexity is $O(N)$, but that is hardly ever hit. If and when this particular case is encountered, the next few cases are guaranteed to have a worst case complexity of $O(1)$! That's why this function performs very well on average. You can consider this function to have a complexity referred to commonly as: Amortized Constant time.

Definition at line 827 of file `bitmap_allocator.h`.

References `__gnu_cxx::__detail::__bit_allocate()`, `__gnu_cxx::__detail::__num_bitmaps()`, and `__gnu_cxx::_Bit_scan←_forward()`.

5.32.2.2 `template<typename _Tp> void __gnu_cxx::bitmap_allocator<_Tp>::_M_deallocate_single_object (pointer __p) throw) [inline]`

Deallocates memory that belongs to a single object of size `sizeof(_Tp)`.

Complexity: $O(\lg(N))$, but the worst case is not hit often! This is because containers usually deallocate memory close to each other and this case is handled in $O(1)$ time by the deallocate function.

Definition at line 917 of file `bitmap_allocator.h`.

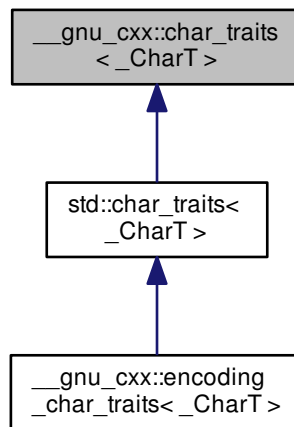
References `std::__addressof()`, `__gnu_cxx::__detail::__bit_free()`, and `__gnu_cxx::__detail::__num_bitmaps()`.

The documentation for this class was generated from the following file:

- [bitmap_allocator.h](#)

5.33 `__gnu_cxx::char_traits<_CharT>` Struct Template Reference

Inheritance diagram for `__gnu_cxx::char_traits<_CharT>`:



Public Types

- `typedef _CharT char_type`
- `typedef _Char_types<_CharT>::int_type int_type`
- `typedef _Char_types<_CharT>::off_type off_type`
- `typedef _Char_types<_CharT>::pos_type pos_type`
- `typedef _Char_types<_CharT>::state_type state_type`

Static Public Member Functions

- static void **assign** (char_type &__c1, const char_type &__c2)
- static char_type * **assign** (char_type *__s, std::size_t __n, char_type __a)
- static int **compare** (const char_type *__s1, const char_type *__s2, std::size_t __n)
- static char_type * **copy** (char_type *__s1, const char_type *__s2, std::size_t __n)
- static constexpr int_type **eof** ()
- static constexpr bool **eq** (const char_type &__c1, const char_type &__c2)
- static constexpr bool **eq_int_type** (const int_type &__c1, const int_type &__c2)
- static const char_type * **find** (const char_type *__s, std::size_t __n, const char_type &__a)
- static std::size_t **length** (const char_type *__s)
- static constexpr bool **lt** (const char_type &__c1, const char_type &__c2)
- static char_type * **move** (char_type *__s1, const char_type *__s2, std::size_t __n)
- static constexpr int_type **not_eof** (const int_type &__c)
- static constexpr char_type **to_char_type** (const int_type &__c)
- static constexpr int_type **to_int_type** (const char_type &__c)

5.33.1 Detailed Description

```
template<typename _CharT>
struct __gnu_cxx::char_traits<_CharT>
```

Base class used to implement `std::char_traits`.

Note

For any given actual character type, this definition is probably wrong. (Most of the member functions are likely to be right, but the `int_type` and `state_type` typedefs, and the `eof()` member function, are likely to be wrong.) The reason this class exists is so users can specialize it. Classes in namespace `std` may not be specialized for fundamental types, but classes in namespace `__gnu_cxx` may be.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/strings.html#strings.string.character_types for advice on how to make use of this class for *unusual* character types. Also, check out `include/ext/pod_char_traits.h`.

Definition at line 83 of file `char_traits.h`.

The documentation for this struct was generated from the following file:

- [char_traits.h](#)

5.34 `__gnu_cxx::character<_Value, _Int, _St>` Struct Template Reference

Public Types

- typedef `character<_Value, _Int, _St>` **char_type**
- typedef `_Int` **int_type**
- typedef `_St` **state_type**
- typedef `_Value` **value_type**

Static Public Member Functions

- `template<typename V2 >`
static `char_type` `from` (const V2 &v)
- `template<typename V2 >`
static V2 `to` (const `char_type` &c)

Public Attributes

- `value_type` `value`

5.34.1 Detailed Description

```
template<typename _Value, typename _Int, typename _St = std::mbstate_t>
struct __gnu_cxx::character< _Value, _Int, _St >
```

A POD class that serves as a character abstraction class.

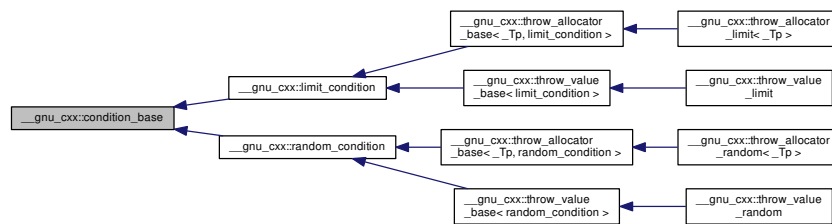
Definition at line 49 of file `pod_char_traits.h`.

The documentation for this struct was generated from the following file:

- [pod_char_traits.h](#)

5.35 __gnu_cxx::condition_base Struct Reference

Inheritance diagram for `__gnu_cxx::condition_base`:



5.35.1 Detailed Description

Base struct for condition policy.

Requires a public member function with the signature `void throw_conditionally()`

Definition at line 403 of file `throw_allocator.h`.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

5.36 `__gnu_cxx::constant_binary_fun<_Result, _Arg1, _Arg2>` Struct Template Reference

Inherits `__gnu_cxx::Constant_binary_fun<_Result, _Arg1, _Arg2>`.

Public Types

- typedef `_Arg1` **first_argument_type**
- typedef `_Result` **result_type**
- typedef `_Arg2` **second_argument_type**

Public Member Functions

- **constant_binary_fun** (const `_Result` &__v)
- const `result_type` & **operator()** (const `_Arg1` &, const `_Arg2` &) const

Public Attributes

- `_Result` **_M_val**

5.36.1 Detailed Description

```
template<class _Result, class _Arg1 = _Result, class _Arg2 = _Arg1>
struct __gnu_cxx::constant_binary_fun<_Result, _Arg1, _Arg2>
```

An [SGI extension](#) .

Definition at line 320 of file `ext/functional`.

The documentation for this struct was generated from the following file:

- [ext/functional](#)

5.37 `__gnu_cxx::constant_unary_fun<_Result, _Argument>` Struct Template Reference

Inherits `__gnu_cxx::Constant_unary_fun<_Result, _Argument>`.

Public Types

- typedef `_Argument` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- **constant_unary_fun** (const _Result &__v)
- const result_type & **operator()** (const _Argument &) const

Public Attributes

- result_type **_M_val**

5.37.1 Detailed Description

```
template<class _Result, class _Argument = _Result>
struct __gnu_cxx::constant_unary_fun< _Result, _Argument >
```

An [SGI extension](#) .

Definition at line 312 of file ext/functional.

The documentation for this struct was generated from the following file:

- [ext/functional](#)

5.38 __gnu_cxx::constant_void_fun< _Result > Struct Template Reference

Inherits [__gnu_cxx::Constant_void_fun< _Result >](#).

Public Types

- typedef _Result **result_type**

Public Member Functions

- **constant_void_fun** (const _Result &__v)
- const result_type & **operator()** () const

Public Attributes

- result_type **_M_val**

5.38.1 Detailed Description

```
template<class _Result>
struct __gnu_cxx::constant_void_fun< _Result >
```

An [SGI extension](#) .

Definition at line 303 of file ext/functional.

The documentation for this struct was generated from the following file:

- [ext/functional](#)

5.39 `__gnu_cxx::debug_allocator<_Alloc>` Class Template Reference

Public Types

- typedef `_Traits::const_pointer` **const_pointer**
- typedef `_Traits::const_reference` **const_reference**
- typedef `_Traits::difference_type` **difference_type**
- typedef `_Traits::pointer` **pointer**
- typedef `_Traits::reference` **reference**
- typedef `_Traits::size_type` **size_type**
- typedef `_Traits::value_type` **value_type**

Public Member Functions

- template<typename `_Alloc2`>
debug_allocator (const [debug_allocator](#)< `_Alloc2` > &__a2, typename `__convertible<_Alloc2>::__type`=0)
- **debug_allocator** (const `_Alloc` &__a)
- pointer **allocate** (size_type __n)
- pointer **allocate** (size_type __n, const void * __hint)
- void **construct** (pointer __p, const value_type &__val)
- template<typename `_Tp`, typename... `_Args`>
void **construct** (`_Tp` * __p, `_Args` &&... __args)
- void **deallocate** (pointer __p, size_type __n)
- template<typename `_Tp`>
void **destroy** (`_Tp` * __p)
- size_type **max_size** () const throw ()

Friends

- template<typename >
class **debug_allocator**
- bool **operator==** (const [debug_allocator](#) &__lhs, const [debug_allocator](#) &__rhs)

5.39.1 Detailed Description

```
template<typename _Alloc>
class __gnu_cxx::debug_allocator< _Alloc >
```

A meta-allocator with debugging bits.

This is precisely the allocator defined in the C++03 Standard.

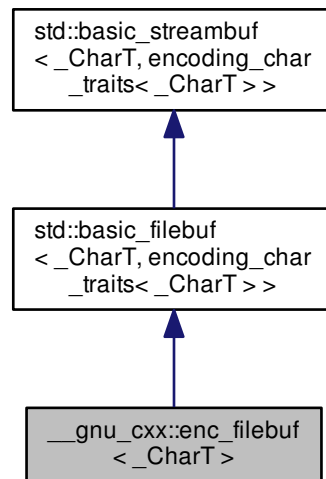
Definition at line 62 of file debug_allocator.h.

The documentation for this class was generated from the following file:

- [debug_allocator.h](#)

5.40 __gnu_cxx::enc_filebuf< _CharT > Class Template Reference

Inheritance diagram for __gnu_cxx::enc_filebuf< _CharT >:



Public Types

- typedef `codecvt< char_type, char, __state_type >` **__codecvt_type**
- typedef `__basic_file< char >` **__file_type**
- typedef `basic_filebuf< char_type, traits_type >` **__filebuf_type**
- typedef `traits_type::state_type` **__state_type**
- typedef `basic_streambuf< char_type, traits_type >` **__streambuf_type**
- typedef `_CharT` **char_type**
- typedef `traits_type::int_type` **int_type**
- typedef `traits_type::off_type` **off_type**
- typedef `traits_type::pos_type` **pos_type**
- typedef `traits_type::state_type` **state_type**
- typedef `encoding_char_traits< _CharT >` **traits_type**

Public Member Functions

- **enc_filebuf** (state_type &__state)
- `__filebuf_type` * **close** ()
- locale **getloc** () const
- streamsize **in_avail** ()
- bool **is_open** () const throw ()
- `__filebuf_type` * **open** (const char * __s, ios_base::openmode __mode)
- `__filebuf_type` * **open** (const std::string & __s, ios_base::openmode __mode)
- locale **pubimbue** (const locale & __loc)
- int_type **sbumpc** ()
- int_type **sgetc** ()
- streamsize **sgetn** (char_type * __s, streamsize __n)
- int_type **snextc** ()
- int_type **sputbackc** (char_type __c)
- int_type **sputc** (char_type __c)
- streamsize **sputn** (const char_type * __s, streamsize __n)
- int_type **sungetc** ()
- void **swap** (basic_filebuf &)
- `basic_streambuf` * **pubsetbuf** (char_type * __s, streamsize __n)
- pos_type **pubseekoff** (off_type __off, ios_base::seekdir __way, ios_base::openmode __mode=ios_base::in|ios_base::out)
- pos_type **pubseekpos** (pos_type __sp, ios_base::openmode __mode=ios_base::in|ios_base::out)
- int **pubsync** ()

Protected Member Functions

- void **__safe_gbump** (streamsize __n)
- void **__safe_pbump** (streamsize __n)
- void **_M_allocate_internal_buffer** ()
- bool **_M_convert_to_external** (char_type *, streamsize)
- void **_M_create_pback** ()
- void **_M_destroy_internal_buffer** () throw ()
- void **_M_destroy_pback** () throw ()
- int **_M_get_ext_pos** (__state_type & __state)
- pos_type **_M_seek** (off_type __off, ios_base::seekdir __way, __state_type __state)
- void **_M_set_buffer** (streamsize __off)
- bool **_M_terminate_output** ()
- void **gbump** (int __n)
- virtual void **imbue** (const locale & __loc)
- virtual int_type **overflow** (int_type __c=encoding_char_traits<_CharT>::eof())
- virtual int_type **pbackfail** (int_type __c=encoding_char_traits<_CharT>::eof())
- void **pbump** (int __n)
- virtual pos_type **seekoff** (off_type __off, ios_base::seekdir __way, ios_base::openmode __mode=ios_base::in|ios_base::out)
- virtual pos_type **seekpos** (pos_type __pos, ios_base::openmode __mode=ios_base::in|ios_base::out)
- virtual `__streambuf_type` * **setbuf** (char_type * __s, streamsize __n)
- void **setg** (char_type * __gbeg, char_type * __gnext, char_type * __gend)
- void **setp** (char_type * __pbeg, char_type * __pend)

- virtual streamsize **showmanyc** ()
- void **swap** (basic_streambuf &__sb)
- virtual int **sync** ()
- virtual int_type **uflow** ()
- virtual int_type **underflow** ()
- virtual streamsize **xsggetn** (char_type *__s, streamsize __n)
- virtual streamsize **xspu** (const char_type *__s, streamsize __n)

- char_type * **eback** () const
- char_type * **gptr** () const
- char_type * **egptr** () const

- char_type * **pbase** () const
- char_type * **pptr** () const
- char_type * **epptr** () const

Protected Attributes

- char_type * **_M_buf**
 - bool **_M_buf_allocated**
 - locale **_M_buf_locale**
 - size_t **_M_buf_size**
 - const __codecvt_type * **_M_codecvt**
 - char * **_M_ext_buf**
 - streamsize **_M_ext_buf_size**
 - char * **_M_ext_end**
 - const char * **_M_ext_next**
 - __file_type **_M_file**
 - char_type * **_M_in_beg**
 - char_type * **_M_in_cur**
 - char_type * **_M_in_end**
 - __c_lock **_M_lock**
 - ios_base::openmode **_M_mode**
 - char_type * **_M_out_beg**
 - char_type * **_M_out_cur**
 - char_type * **_M_out_end**
 - bool **_M_reading**
 - __state_type **_M_state_beg**
 - __state_type **_M_state_cur**
 - __state_type **_M_state_last**
 - bool **_M_writing**
-
- char_type **_M_pback**
 - char_type * **_M_pback_cur_save**
 - char_type * **_M_pback_end_save**
 - bool **_M_pback_init**

5.40.1 Detailed Description

```
template<typename _CharT>
class __gnu_cxx::enc_filebuf<_CharT>
```

class `enc_filebuf`.

Definition at line 42 of file `enc_filebuf.h`.

5.40.2 Member Function Documentation

5.40.2.1 `void std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_create_pback()` `[inline]`, `[protected]`, `[inherited]`

Initializes pback buffers, and moves normal buffers to safety. Assumptions: `_M_in_cur` has already been moved back

Definition at line 191 of file `fstream`.

5.40.2.2 `void std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_destroy_pback()` `throw` `[inline]`, `[protected]`, `[inherited]`

Deactivates pback buffer contents, and restores normal buffer. Assumptions: The pback buffer has only moved forward.

Definition at line 208 of file `fstream`.

5.40.2.3 `void std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_set_buffer(streamsize __off)` `[inline]`, `[protected]`, `[inherited]`

This function sets the pointers of the internal buffer, both get and put areas. Typically:

`__off == egptr() - eback()` upon underflow/uflow (**read** mode); `__off == 0` upon overflow (**write** mode); `__off == -1` upon open, setbuf, seekoff/pos (**uncommitted** mode).

NB: `egptr() - pbase() == _M_buf_size - 1`, since `_M_buf_size` reflects the actual allocated memory and the last cell is reserved for the overflow char of a full put area.

Definition at line 422 of file `fstream`.

5.40.2.4 `__filebuf_type* std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::close()` `[inherited]`

Closes the currently associated file.

Returns

`this` on success, `NULL` on failure

If no file is currently open, this function immediately fails.

If a *put buffer area* exists, `overflow(eof)` is called to flush all the characters. The file is then closed.

If any operations fail, this function also fails.

5.40.2.5 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::eback ()`
`const [inline], [protected], [inherited]`

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Definition at line 482 of file `streambuf`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::imbue()`, `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, `std::basic_filebuf< _CharT, _Traits >::seekpos()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

5.40.2.6 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::egptr ()`
`const [inline], [protected], [inherited]`

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Definition at line 488 of file `streambuf`.

Referenced by `std::ostreambuf_iterator< _CharT, _Traits >::failed()`, `std::basic_filebuf< _CharT, _Traits >::seekpos()`, `std::basic_filebuf< _CharT, _Traits >::showmanyc()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

5.40.2.7 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::epptr ()`
`const [inline], [protected], [inherited]`

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

Definition at line 535 of file `streambuf`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::xspn()`.

5.40.2.8 `template<typename _CharT, typename _Traits> void std::basic_streambuf< _CharT, _Traits >::gbump (int __n)`
`[inline], [protected], [inherited]`

Moving the read position.

Parameters

<code>_↔</code>	The delta by which to move.
<code>_n</code>	

This just advances the read position without returning any data.

Definition at line 498 of file `streambuf`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::pbackfail()`, and `std::basic_filebuf<_CharT, _Traits>::xsgetn()`.

5.40.2.9 `template<typename _CharT, typename _Traits> locale std::basic_streambuf<_CharT, _Traits>::getloc () const`
`[inline], [inherited]`

Locale access.

Returns

The current locale in effect.

If `pubimbue(loc)` has been called, then the most recent `loc` is returned. Otherwise the global locale in effect at the time of construction is returned.

Definition at line 226 of file `streambuf`.

5.40.2.10 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::gptr ()`
`const [inline], [protected], [inherited]`

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Definition at line 485 of file `streambuf`.

Referenced by `std::ostreambuf_iterator<_CharT, _Traits>::failed()`, `std::basic_filebuf<_CharT, _Traits>::imbue()`, `std::basic_filebuf<_CharT, _Traits>::pbackfail()`, `std::basic_filebuf<_CharT, _Traits>::seekpos()`, `std::basic_filebuf<_CharT, _Traits>::showmanyc()`, `std::basic_filebuf<_CharT, _Traits>::underflow()`, and `std::basic_filebuf<_CharT, _Traits>::xsgetn()`.

5.40.2.11 `virtual void std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::imbue (const locale & __loc)`
`[protected], [virtual], [inherited]`

Changes translations.

Parameters

<code>__loc</code>	A new locale.
--------------------	---------------

Translations done during I/O which depend on the current locale are changed by this call. The standard adds, *Between invocations of this function a class derived from streambuf can safely cache results of calls to locale functions and to members of facets so obtained.*

Note

Base class version does nothing.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

5.40.2.12 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf<_CharT, _Traits>::in_avail ()`
`[inline], [inherited]`

Looking ahead into the stream.

Returns

The number of characters available.

If a read position is available, returns the number of characters available for reading before the buffer must be refilled. Otherwise returns the derived `showmanyc()`.

Definition at line 284 of file streambuf.

5.40.2.13 `bool std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::is_open () const throw` `[inline], [inherited]`

Returns true if the external file is open.

Definition at line 252 of file fstream.

5.40.2.14 `__filebuf_type* std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::open (const char * __s,`
`ios_base::openmode __mode)` `[inherited]`

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Returns

`this` on success, NULL on failure

If a file is already open, this function immediately fails. Otherwise it tries to open the file named `__s` using the flags given in `__mode`.

Table 92, adapted here, gives the relation between openmode combinations and the equivalent `fopen()` flags. (NB: lines `app`, `in|out|app`, `in|app`, `binary|app`, `binary|in|out|app`, and `binary|in|app` per DR 596)

ios_base Flag combination					stdio equivalent
binary	in	out	trunc	app	
		+			w
		+		+	a
				+	a
		+	+		w
	+				r
	+	+			r+
	+	+	+		w+
	+	+		+	a+
	+			+	a+
+		+			wb
+		+		+	ab
+				+	ab
+		+	+		wb
+	+				rb
+	+	+			r+b
+	+	+	+		w+b
+	+	+		+	a+b
+	+			+	a+b

5.40.2.15 `__filebuf_type* std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::open (const std::string & __s, ios_base::openmode __mode)` `[inline]`, `[inherited]`

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Returns

`this` on success, NULL on failure

Definition at line 307 of file `fstream`.

5.40.2.16 `virtual int_type std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::overflow (int_type __c = _Traits::eof()) [protected], [virtual], [inherited]`

Consumes data from the buffer; writes to the controlled sequence.

Parameters

<code>__c</code>	An additional character to consume.
------------------	-------------------------------------

Returns

`eof()` to indicate failure, something else (usually `__c`, or `not_eof()`)

Informally, this function is called when the output buffer is full (or does not exist, as buffering need not actually be done). If a buffer exists, it is *consumed*, with *some effect* on the controlled sequence. (Typically, the buffer is written out to the sequence verbatim.) In either case, the character `c` is also written out, if `__c` is not `eof()`.

For a formal definition of this function, see a good text such as Langer & Kreft, or [27.5.2.4.5]/3-7.

A functioning output streambuf can be created by overriding only this function (no buffer area will be used).

Note

Base class version does nothing, returns `eof()`.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

5.40.2.17 `virtual int_type std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::pbackfail (int_type __c = _Traits::eof()) [protected], [virtual], [inherited]`

Tries to back up the input sequence.

Parameters

<code>__c</code>	The character to be inserted back into the sequence.
------------------	--

Returns

`eof()` on failure, *some other value* on success

Postcondition

The constraints of `gptr()`, `eback()`, and `pptr()` are the same as for `underflow()`.

Note

Base class version does nothing, returns `eof()`.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

5.40.2.18 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::pbase ()`
`const [inline], [protected], [inherited]`

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

Definition at line 529 of file `streambuf`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_filebuf<_CharT, _Traits>::seekoff()`, `std::basic_filebuf<_CharT, _Traits>::seekpos()`, `std::basic_filebuf<_CharT, _Traits>::sync()`, and `std::basic_filebuf<_CharT, _Traits>::xsputn()`.

5.40.2.19 `template<typename _CharT, typename _Traits> void std::basic_streambuf<_CharT, _Traits>::pbump (int __n)`
`[inline], [protected], [inherited]`

Moving the write position.

Parameters

<code>__n</code>	The delta by which to move.
------------------	-----------------------------

This just advances the write position without returning any data.

Definition at line 545 of file `streambuf`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::overflow()`.

5.40.2.20 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::pptr ()`
`const [inline], [protected], [inherited]`

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

Definition at line 532 of file `streambuf`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_filebuf<_CharT, _Traits>::seekoff()`, `std::basic_filebuf<_CharT, _Traits>::seekpos()`, `std::basic_filebuf<_CharT, _Traits>::sync()`, and `std::basic_filebuf<_CharT, _Traits>::xsputn()`.

5.40.2.21 `template<typename _CharT, typename _Traits> locale std::basic_streambuf<_CharT, _Traits>::pubimbue (const locale & __loc) [inline], [inherited]`

Entry point for imbue().

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls the derived imbue(__loc).

Definition at line 209 of file streambuf.

5.40.2.22 `template<typename _CharT, typename _Traits> pos_type std::basic_streambuf<_CharT, _Traits>::pubseekoff (off_type __off, ios_base::seekdir __way, ios_base::openmode __mode = ios_base::in | ios_base::out) [inline], [inherited]`

Alters the stream position.

Parameters

<code>__off</code>	Offset.
<code>__way</code>	Value for ios_base::seekdir.
<code>__mode</code>	Value for ios_base::openmode.

Calls virtual seekoff function.

Definition at line 251 of file streambuf.

5.40.2.23 `template<typename _CharT, typename _Traits> pos_type std::basic_streambuf<_CharT, _Traits>::pubseekpos (pos_type __sp, ios_base::openmode __mode = ios_base::in | ios_base::out) [inline], [inherited]`

Alters the stream position.

Parameters

<code>__sp</code>	Position
<code>__mode</code>	Value for ios_base::openmode.

Calls virtual seekpos function.

Definition at line 263 of file streambuf.

5.40.2.24 `template<typename _CharT, typename _Traits> basic_streambuf* std::basic_streambuf<_CharT, _Traits>::pubsetbuf(char_type* __s, streamsize __n)` `[inline]`, `[inherited]`

Entry points for derived buffer functions.

The public versions of `pubfoo` dispatch to the protected derived `foo` member functions, passing the arguments (if any) and returning the result unchanged.

Definition at line 239 of file `streambuf`.

5.40.2.25 `template<typename _CharT, typename _Traits> int std::basic_streambuf<_CharT, _Traits>::pubsync()` `[inline]`, `[inherited]`

Calls virtual sync function.

Definition at line 271 of file `streambuf`.

Referenced by `std::basic_istream<_CharT, _Traits>::sync()`.

5.40.2.26 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::sbumpc()` `[inline]`, `[inherited]`

Getting the next character.

Returns

The next character, or `eof`.

If the input read position is available, returns that character and increments the read pointer, otherwise calls and returns `uflow()`.

Definition at line 316 of file `streambuf`.

Referenced by `std::basic_istream<_CharT, _Traits>::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, and `std::istreambuf_iterator<_CharT, _Traits>::operator++()`.

5.40.2.27 `virtual pos_type std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::seekoff(off_type __off, ios_base::seekdir __way, ios_base::openmode __mode = ios_base::in | ios_base::out)` `[protected]`, `[virtual]`, `[inherited]`

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

5.40.2.28 `virtual pos_type std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::seekpos (pos_type __pos, ios_base::openmode __mode = ios_base::in | ios_base::out) [protected], [virtual], [inherited]`

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

5.40.2.29 `virtual __streambuf_type* std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::setbuf (char_type* __s, streamsize __n) [protected], [virtual], [inherited]`

Manipulates the buffer.

Parameters

<code>__s</code>	Pointer to a buffer area.
<code>__n</code>	Size of <code>__s</code> .

Returns

`this`

If no file has been opened, and both `__s` and `__n` are zero, then the stream becomes unbuffered. Otherwise, `__s` is used as a buffer; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html#io.streambuf.buffering> for more.

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

5.40.2.30 `template<typename _CharT, typename _Traits> void std::basic_streambuf<_CharT, _Traits>::setg (char_type* __gbeg, char_type* __gnext, char_type* __gend) [inline], [protected], [inherited]`

Setting the three read area pointers.

Parameters

<code>__gbeg</code>	A pointer.
<code>__gnext</code>	A pointer.
<code>__gend</code>	A pointer.

Postcondition

`__gbeg == eback()`, `__gnext == gptr()`, and `__gend == egptr()`

Definition at line 509 of file `streambuf`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::xsgetn()`.

5.40.231 `template<typename _CharT, typename _Traits> void std::basic_streambuf<_CharT, _Traits>::setp(char_type * __pbeg, char_type * __pend)` `[inline]`, `[protected]`, `[inherited]`

Setting the three write area pointers.

Parameters

<code>__pbeg</code>	A pointer.
<code>__pend</code>	A pointer.

Postcondition

`__pbeg == pbase()`, `__pbeg == pptr()`, and `__pend == epptr()`

Definition at line 555 of file `streambuf`.

5.40.232 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::sgetc()` `[inline]`, `[inherited]`

Getting the next character.

Returns

The next character, or `eof`.

If the input read position is available, returns that character, otherwise calls and returns `underflow()`. Does not move the read position after fetching the character.

Definition at line 338 of file `streambuf`.

Referenced by `std::istreambuf_iterator<_CharT, _Traits>::equal()`, `std::ostreambuf_iterator<_CharT, _Traits>::failed()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, and `std::basic_streambuf<_CharT, _Traits>::xsputn()`.

5.40.233 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf<_CharT, _Traits>::sgetn(char_type * __s, streamsize __n)` `[inline]`, `[inherited]`

Entry point for `xsgetn`.

Parameters

\leftrightarrow __s	A buffer area.
\leftrightarrow __n	A count.

Returns `xsgetn(__s,__n)`. The effect is to fill `__s[0]` through `__s[__n-1]` with characters from the input sequence, if possible.

Definition at line 357 of file `streambuf`.

5.40.2.34 `virtual streamsize std::basic_filebuf<_CharT,encoding_char_traits<_CharT>>::showmanyc ()`
`[protected],[virtual],[inherited]`

Investigating the data available.

Returns

An estimate of the number of characters available in the input sequence, or -1.

If it returns a positive value, then successive calls to `underflow()` will not return `traits::eof()` until at least that number of characters have been supplied. If `showmanyc()` returns -1, then calls to `underflow()` or `uflow()` will fail. [27.5.2.4.3]/1

Note

Base class version does nothing, returns zero.

The standard adds that *the intention is not only that the calls [to `underflow` or `uflow`] will not return `eof()` but that they will return immediately.*

The standard adds that *the morphemes of `showmanyc` are **es-how-many-see**, not **show-manic**.*

Reimplemented from `std::basic_streambuf<_CharT,_Traits>`.

5.40.2.35 `template<typename _CharT,typename _Traits> int_type std::basic_streambuf<_CharT,_Traits>::snextc ()`
`[inline],[inherited]`

Getting the next character.

Returns

The next character, or `eof`.

Calls `sbumpc()`, and if that function returns `traits::eof()`, so does this function. Otherwise, `sgetc()`.

Definition at line 298 of file `streambuf`.

Referenced by `std::ostreambuf_iterator<_CharT,_Traits>::failed()`, `std::basic_istream<_CharT,_Traits>::get()`, `std::basic_istream<_CharT,_Traits>::getline()`, `std::basic_istream<_CharT,_Traits>::ignore()`, `std::basic_istream<_CharT,_Traits>::sentry::sentry()`, and `std::basic_streambuf<_CharT,_Traits>::xsputn()`.

5.40.2.36 `template<typename _CharT,typename _Traits> int_type std::basic_streambuf<_CharT,_Traits>::sputbackc (char_type _c)` `[inline],[inherited]`

Pushing characters back into the input stream.

Parameters

<code>__c</code>	The character to push back.
------------------	-----------------------------

Returns

The previous character, if possible.

Similar to `sungetc()`, but `__c` is pushed onto the stream instead of *the previous character*. If successful, the next character fetched from the input stream will be `__c`.

Definition at line 372 of file `streambuf`.

Referenced by `std::basic_istream<_CharT, _Traits>::putback()`.

5.40.2.37 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::sputc (char_type __c) [inline], [inherited]`

Entry point for all single-character output functions.

Parameters

<code>__c</code>	A character to output.
------------------	------------------------

Returns

`__c`, if possible.

One of two public output functions.

If a write position is available for the output sequence (i.e., the buffer is not full), stores `__c` in that position, increments the position, and returns `traits::to_int_type(__c)`. If a write position is not available, returns `overflow(__c)`.

Definition at line 424 of file `streambuf`.

Referenced by `std::basic_istream<_CharT, _Traits>::get()`, `std::ostreambuf_iterator<_CharT, _Traits>::operator=()`, and `std::basic_streambuf<_CharT, _Traits>::xsputn()`.

5.40.2.38 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf<_CharT, _Traits>::sputn (const char_type * __s, streamsize __n) [inline], [inherited]`

Entry point for all single-character output functions.

Parameters

<code>__s</code>	A buffer read area.
<code>__n</code>	A count.

One of two public output functions.

Returns `xspn(__s,__n)`. The effect is to write `__s[0]` through `__s[__n-1]` to the output sequence, if possible.

Definition at line 450 of file `streambuf`.

Referenced by `std::ostreambuf_iterator<_CharT, _Traits >::failed()`.

5.40.2.39 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits >::sungetc ()`
`[inline], [inherited]`

Moving backwards in the input stream.

Returns

The previous character, if possible.

If a putback position is available, this function decrements the input pointer and returns that character. Otherwise, calls and returns `pbackfail()`. The effect is to *unget* the last character *gotten*.

Definition at line 397 of file `streambuf`.

Referenced by `std::basic_istream<_CharT, _Traits >::unget()`.

5.40.2.40 `virtual int std::basic_filebuf<_CharT, encoding_char_traits<_CharT > >::sync (void)` `[protected],`
`[virtual], [inherited]`

Synchronizes the buffer arrays with the controlled sequences.

Returns

-1 on failure.

Each derived class provides its own appropriate behavior, including the definition of *failure*.

Note

Base class version does nothing, returns zero.

Reimplemented from [std::basic_streambuf<_CharT, _Traits >](#).

5.40.2.41 `template<typename _CharT, typename _Traits> virtual int_type std::basic_streambuf<_CharT, _Traits>::uflow ()` `[inline], [protected], [virtual], [inherited]`

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function does the same thing as `underflow()`, and in fact is required to call that function. It also returns the new character, like `underflow()` does. However, this function also moves the read position forward by one.

Reimplemented in `__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>`.

Definition at line 700 of file `streambuf`.

5.40.2.42 `virtual int_type std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::underflow ()` `[protected], [virtual], [inherited]`

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function is called when the input buffer is exhausted (or does not exist, as buffering need not actually be done). If a buffer exists, it is *refilled*. In either case, the next available character is returned, or `traits::eof()` to indicate a null pending sequence.

For a formal definition of the pending sequence, see a good text such as Langer & Kreft, or [27.5.2.4.3]/7-14.

A functioning input streambuf can be created by overriding only this function (no buffer area will be used). For an example, see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html>

Note

Base class version does nothing, returns `eof()`.

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

5.40.2.43 `virtual streamsize std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::xsgetn (char_type * __s, streamsize __n)` `[protected], [virtual], [inherited]`

Multiple character extraction.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to assign.

Returns

The number of characters assigned.

Fills `__s[0]` through `__s[__n-1]` with characters from the input sequence, as if by `sbumpc()`. Stops when either `__n` characters have been copied, or when `traits::eof()` would be copied.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

5.40.2.44 `virtual streamsize std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::xsputn (const char_type * __s, streamsize __n)` [protected], [virtual], [inherited]

Multiple character insertion.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to write.

Returns

The number of characters written.

Writes `__s[0]` through `__s[__n-1]` to the output sequence, as if by `sputc()`. Stops when either `n` characters have been copied, or when `sputc()` would return `traits::eof()`.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

5.40.3 Member Data Documentation

5.40.3.1 `char_type* std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_buf` [protected], [inherited]

Pointer to the beginning of internal buffer.

Definition at line 128 of file `fstream`.

5.40.3.2 `template<typename _CharT, typename _Traits> locale std::basic_streambuf<_CharT, _Traits>::_M_buf_locale`
`[protected], [inherited]`

Current locale setting.

Definition at line 192 of file `streambuf`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::basic_filebuf()`.

5.40.3.3 `size_t std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_buf_size` `[protected],`
`[inherited]`

Actual size of internal buffer. This number is equal to the size of the put area + 1 position, reserved for the overflow char of a full area.

Definition at line 135 of file `fstream`.

5.40.3.4 `char* std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_ext_buf` `[protected],`
`[inherited]`

Buffer for external characters. Used for input when `codecvt::always_noconv() == false`. When valid, this corresponds to `eback()`.

Definition at line 170 of file `fstream`.

5.40.3.5 `streamsize std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_ext_buf_size`
`[protected], [inherited]`

Size of buffer held by `_M_ext_buf`.

Definition at line 175 of file `fstream`.

5.40.3.6 `const char* std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_ext_next` `[protected],`
`[inherited]`

Pointers into the buffer held by `_M_ext_buf` that delimit a subsequence of bytes that have been read but not yet converted. When valid, `_M_ext_next` corresponds to `egptr()`.

Definition at line 182 of file `fstream`.

5.40.3.7 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::_M_in_beg`
`[protected], [inherited]`

Start of get area.

Definition at line 184 of file `streambuf`.

5.40.3.8 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::_M_in_cur
[protected], [inherited]`

Current read area.

Definition at line 185 of file streambuf.

5.40.3.9 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::_M_in_end
[protected], [inherited]`

End of get area.

Definition at line 186 of file streambuf.

5.40.3.10 `ios_base::openmode std::basic_filebuf< _CharT, encoding_char_traits< _CharT > >::_M_mode
[protected], [inherited]`

Place to stash in || out || in | out settings for current filebuf.

Definition at line 113 of file fstream.

5.40.3.11 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits
>::_M_out_beg [protected], [inherited]`

Start of put area.

Definition at line 187 of file streambuf.

5.40.3.12 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_cur
[protected], [inherited]`

Current put area.

Definition at line 188 of file streambuf.

5.40.3.13 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits
>::_M_out_end [protected], [inherited]`

End of put area.

Definition at line 189 of file streambuf.

5.40.3.14 `char_type std::basic_filebuf< _CharT, encoding_char_traits< _CharT > >::_M_pback [protected],
[inherited]`

Necessary bits for putback buffer management.

Note

pbacks of over one character are not currently supported.

Definition at line 156 of file fstream.

5.40.3.15 `char_type* std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_pback_cur_save`
[protected], [inherited]

Necessary bits for putback buffer management.

Note

pbacks of over one character are not currently supported.

Definition at line 157 of file `fstream`.

5.40.3.16 `char_type* std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_pback_end_save`
[protected], [inherited]

Necessary bits for putback buffer management.

Note

pbacks of over one character are not currently supported.

Definition at line 158 of file `fstream`.

5.40.3.17 `bool std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_pback_init` [protected],
[inherited]

Necessary bits for putback buffer management.

Note

pbacks of over one character are not currently supported.

Definition at line 159 of file `fstream`.

5.40.3.18 `bool std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>::_M_reading` [protected],
[inherited]

`_M_reading == false && _M_writing == false` for **uncommitted** mode; `_M_reading == true` for **read** mode; `_M_writing == true` for **write** mode;

NB: `_M_reading == true && _M_writing == true` is unused.

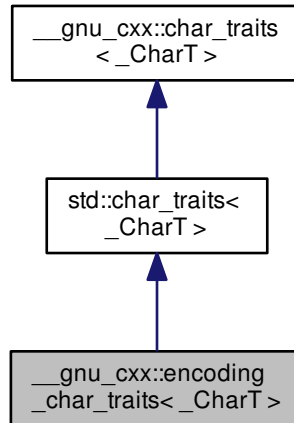
Definition at line 147 of file `fstream`.

The documentation for this class was generated from the following file:

- [enc_filebuf.h](#)

5.41 `__gnu_cxx::encoding_char_traits<_CharT>` Struct Template Reference

Inheritance diagram for `__gnu_cxx::encoding_char_traits<_CharT>`:



Public Types

- typedef `_CharT char_type`
- typedef `_Char_types<_CharT>::int_type int_type`
- typedef `_Char_types<_CharT>::off_type off_type`
- typedef `std::fpos<state_type> pos_type`
- typedef `encoding_state state_type`

Static Public Member Functions

- static void **assign** (`char_type &__c1`, `const char_type &__c2`)
- static `char_type *` **assign** (`char_type *__s`, `std::size_t __n`, `char_type __a`)
- static int **compare** (`const char_type *__s1`, `const char_type *__s2`, `std::size_t __n`)
- static `char_type *` **copy** (`char_type *__s1`, `const char_type *__s2`, `std::size_t __n`)
- static constexpr int_type **eof** ()
- static constexpr bool **eq** (`const char_type &__c1`, `const char_type &__c2`)
- static constexpr bool **eq_int_type** (`const int_type &__c1`, `const int_type &__c2`)
- static const `char_type *` **find** (`const char_type *__s`, `std::size_t __n`, `const char_type &__a`)
- static `std::size_t` **length** (`const char_type *__s`)
- static constexpr bool **lt** (`const char_type &__c1`, `const char_type &__c2`)
- static `char_type *` **move** (`char_type *__s1`, `const char_type *__s2`, `std::size_t __n`)
- static constexpr int_type **not_eof** (`const int_type &__c`)
- static constexpr `char_type` **to_char_type** (`const int_type &__c`)
- static constexpr int_type **to_int_type** (`const char_type &__c`)

5.41.1 Detailed Description

```
template<typename _CharT>
struct __gnu_cxx::encoding_char_traits< _CharT >
```

`encoding_char_traits`

Definition at line 211 of file `codecvt_specializations.h`.

The documentation for this struct was generated from the following file:

- [codecvt_specializations.h](#)

5.42 `__gnu_cxx::encoding_state` Class Reference

Public Types

- typedef `iconv_t` **descriptor_type**

Public Member Functions

- **encoding_state** (const char * __int, const char * __ext, int __ibom=0, int __ebom=0, int __bytes=1)
- **encoding_state** (const [encoding_state](#) & __obj)
- int **character_ratio** () const
- int **external_bom** () const
- const [std::string](#) **external_encoding** () const
- bool **good** () const throw ()
- const `descriptor_type` & **in_descriptor** () const
- int **internal_bom** () const
- const [std::string](#) **internal_encoding** () const
- [encoding_state](#) & **operator=** (const [encoding_state](#) & __obj)
- const `descriptor_type` & **out_descriptor** () const

Protected Member Functions

- void **construct** (const [encoding_state](#) & __obj)
- void **destroy** () throw ()
- void **init** ()

Protected Attributes

- int **_M_bytes**
- int **_M_ext_bom**
- [std::string](#) **_M_ext_enc**
- `descriptor_type` **_M_in_desc**
- int **_M_int_bom**
- [std::string](#) **_M_int_enc**
- `descriptor_type` **_M_out_desc**

5.42.1 Detailed Description

Extension to use iconv for dealing with character encodings.

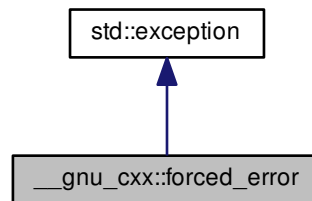
Definition at line 51 of file codecvt_specializations.h.

The documentation for this class was generated from the following file:

- [codecvt_specializations.h](#)

5.43 __gnu_cxx::forced_error Struct Reference

Inheritance diagram for __gnu_cxx::forced_error:



Public Member Functions

- virtual const char * [what](#) () const _GLIBCXX_TXN_SAFE_DYN noexcept

5.43.1 Detailed Description

Thrown by exception safety machinery.

Definition at line 74 of file throw_allocator.h.

5.43.2 Member Function Documentation

5.43.2.1 virtual const char* std::exception::what () const [virtual], [noexcept], [inherited]

Returns a C-style character string describing the general cause of the current error.

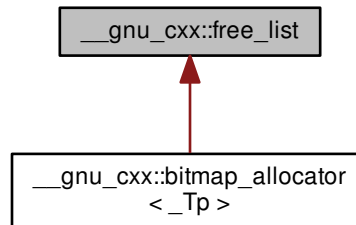
Reimplemented in [std::bad_function_call](#), [std::ios_base::failure](#), [std::runtime_error](#), [std::bad_typeid](#), [std::bad_cast](#), [std::logic_error](#), [std::future_error](#), [std::bad_exception](#), [std::bad_weak_ptr](#), [std::experimental::fundamentals_v1::bad_any_cast](#), and [std::bad_alloc](#).

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

5.44 `__gnu_cxx::free_list` Class Reference

Inheritance diagram for `__gnu_cxx::free_list`:



Public Types

- typedef `__mutex` **`__mutex_type`**
- typedef `vector_type::iterator` **`iterator`**
- typedef `size_t *` **`value_type`**
- typedef `__detail::__mini_vector< value_type >` **`vector_type`**

Public Member Functions

- void `_M_clear` ()
- `size_t *` `_M_get` (`size_t __sz`) throw (`std::bad_alloc`)
- void `_M_insert` (`size_t *__addr`) throw ()

5.44.1 Detailed Description

The free list class for managing chunks of memory to be given to and returned by the `bitmap_allocator`.

Definition at line 521 of file `bitmap_allocator.h`.

5.44.2 Member Function Documentation

5.44.2.1 void `__gnu_cxx::free_list::_M_clear` ()

This function just clears the internal Free List, and gives back all the memory to the OS.

5.44.2.2 `size_t *` `__gnu_cxx::free_list::_M_get` (`size_t __sz`) throw `std::bad_alloc`

This function gets a block of memory of the specified size from the free list.

Parameters

<code>__sz</code>	The size in bytes of the memory required.
-------------------	---

Returns

A pointer to the new memory block of size at least equal to that requested.

5.44.2.3 `void __gnu_cxx::free_list::M_insert (size_t * __addr) throw () [inline]`

This function returns the block of memory to the internal free list.

Parameters

<code>__addr</code>	The pointer to the memory block that was given by a call to the <code>_M_get</code> function.
---------------------	---

Definition at line 631 of file `bitmap_allocator.h`.

The documentation for this class was generated from the following file:

- [bitmap_allocator.h](#)

5.45 `__gnu_cxx::hash_map< _Key, _Tp, _HashFn, _EqualKey, _Alloc >` Class Template Reference

Public Types

- `typedef _Ht::allocator_type allocator_type`
- `typedef _Ht::const_iterator const_iterator`
- `typedef _Ht::const_pointer const_pointer`
- `typedef _Ht::const_reference const_reference`
- `typedef _Tp data_type`
- `typedef _Ht::difference_type difference_type`
- `typedef _Ht::hasher hasher`
- `typedef _Ht::iterator iterator`
- `typedef _Ht::key_equal key_equal`
- `typedef _Ht::key_type key_type`
- `typedef _Tp mapped_type`
- `typedef _Ht::pointer pointer`
- `typedef _Ht::reference reference`
- `typedef _Ht::size_type size_type`
- `typedef _Ht::value_type value_type`

Public Member Functions

- **hash_map** (size_type __n)
- **hash_map** (size_type __n, const hasher &__hf)
- **hash_map** (size_type __n, const hasher &__hf, const key_equal &__eq, const allocator_type &__a=allocator_type())
- template<class _InputIterator >
hash_map (_InputIterator __f, _InputIterator __l)
- template<class _InputIterator >
hash_map (_InputIterator __f, _InputIterator __l, size_type __n)
- template<class _InputIterator >
hash_map (_InputIterator __f, _InputIterator __l, size_type __n, const hasher &__hf)
- template<class _InputIterator >
hash_map (_InputIterator __f, _InputIterator __l, size_type __n, const hasher &__hf, const key_equal &__eq, const allocator_type &__a=allocator_type())
- iterator **begin** ()
- const_iterator **begin** () const
- size_type **bucket_count** () const
- void **clear** ()
- size_type **count** (const key_type &__key) const
- size_type **elems_in_bucket** (size_type __n) const
- bool **empty** () const
- iterator **end** ()
- const_iterator **end** () const
- pair< iterator, iterator > **equal_range** (const key_type &__key)
- pair< const_iterator, const_iterator > **equal_range** (const key_type &__key) const
- size_type **erase** (const key_type &__key)
- void **erase** (iterator __it)
- void **erase** (iterator __f, iterator __l)
- iterator **find** (const key_type &__key)
- const_iterator **find** (const key_type &__key) const
- allocator_type **get_allocator** () const
- hasher **hash_func** () const
- pair< iterator, bool > **insert** (const value_type &__obj)
- template<class _InputIterator >
void **insert** (_InputIterator __f, _InputIterator __l)
- pair< iterator, bool > **insert_noresize** (const value_type &__obj)
- key_equal **key_eq** () const
- size_type **max_bucket_count** () const
- size_type **max_size** () const
- _Tp & **operator[]** (const key_type &__key)
- void **resize** (size_type __hint)
- size_type **size** () const
- void **swap** (hash_map &__hs)

Friends

- template<class _K1, class _T1, class _HF, class _EqK, class _AI >
bool **operator==** (const hash_map< _K1, _T1, _HF, _EqK, _AI > &, const hash_map< _K1, _T1, _HF, _EqK, _AI > &)

5.45.1 Detailed Description

```
template<class _Key, class _Tp, class _HashFn = hash<_Key>, class _EqualKey = equal_to<_Key>, class _Alloc = allocator<_Tp>>
class __gnu_cxx::hash_map< _Key, _Tp, _HashFn, _EqualKey, _Alloc >
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation_style.html

Definition at line 83 of file hash_map.

The documentation for this class was generated from the following file:

- [hash_map](#)

5.46 __gnu_cxx::hash_multimap< _Key, _Tp, _HashFn, _EqualKey, _Alloc > Class Template Reference

Public Types

- typedef _Ht::allocator_type **allocator_type**
- typedef _Ht::const_iterator **const_iterator**
- typedef [_Ht::const_pointer](#) **const_pointer**
- typedef [_Ht::const_reference](#) **const_reference**
- typedef _Tp **data_type**
- typedef _Ht::difference_type **difference_type**
- typedef _Ht::hasher **hasher**
- typedef _Ht::iterator **iterator**
- typedef _Ht::key_equal **key_equal**
- typedef _Ht::key_type **key_type**
- typedef _Tp **mapped_type**
- typedef [_Ht::pointer](#) **pointer**
- typedef [_Ht::reference](#) **reference**
- typedef _Ht::size_type **size_type**
- typedef [_Ht::value_type](#) **value_type**

Public Member Functions

- **hash_multimap** (size_type __n)
- **hash_multimap** (size_type __n, const hasher &__hf)
- **hash_multimap** (size_type __n, const hasher &__hf, const key_equal &__eq, const allocator_type &__a=allocator_type())
- template<class _InputIterator >
hash_multimap (_InputIterator __f, _InputIterator __l)
- template<class _InputIterator >
hash_multimap (_InputIterator __f, _InputIterator __l, size_type __n)
- template<class _InputIterator >
hash_multimap (_InputIterator __f, _InputIterator __l, size_type __n, const hasher &__hf)
- template<class _InputIterator >
hash_multimap (_InputIterator __f, _InputIterator __l, size_type __n, const hasher &__hf, const key_equal &__eq, const allocator_type &__a=allocator_type())
- iterator **begin** ()
- const_iterator **begin** () const
- size_type **bucket_count** () const
- void **clear** ()
- size_type **count** (const key_type &__key) const
- size_type **elems_in_bucket** (size_type __n) const
- bool **empty** () const
- iterator **end** ()
- const_iterator **end** () const
- pair< iterator, iterator > **equal_range** (const key_type &__key)
- pair< const_iterator, const_iterator > **equal_range** (const key_type &__key) const
- size_type **erase** (const key_type &__key)
- void **erase** (iterator __it)
- void **erase** (iterator __f, iterator __l)
- iterator **find** (const key_type &__key)
- const_iterator **find** (const key_type &__key) const
- allocator_type **get_allocator** () const
- hasher **hash_funct** () const
- iterator **insert** (const value_type &__obj)
- template<class _InputIterator >
void **insert** (_InputIterator __f, _InputIterator __l)
- iterator **insert_noresize** (const value_type &__obj)
- key_equal **key_eq** () const
- size_type **max_bucket_count** () const
- size_type **max_size** () const
- void **resize** (size_type __hint)
- size_type **size** () const
- void **swap** (hash_multimap &__hs)

Friends

- template<class _K1, class _T1, class _HF, class _EqK, class _AI >
bool **operator==** (const hash_multimap< _K1, _T1, _HF, _EqK, _AI > &, const hash_multimap< _K1, _T1, _HF, _EqK, _AI > &)

5.46.1 Detailed Description

```
template<class _Key, class _Tp, class _HashFn = hash<_Key>, class _EqualKey = equal_to<_Key>, class _Alloc = allocator<_Tp>>
class __gnu_cxx::hash_multimap<_Key, _Tp, _HashFn, _EqualKey, _Alloc>
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation_style.html

Definition at line 296 of file hash_map.

The documentation for this class was generated from the following file:

- [hash_map](#)

5.47 __gnu_cxx::hash_multiset<_Value, _HashFcn, _EqualKey, _Alloc> Class Template Reference

Public Types

- typedef _Ht::allocator_type **allocator_type**
- typedef _Ht::const_iterator **const_iterator**
- typedef _Alloc::const_pointer **const_pointer**
- typedef _Alloc::const_reference **const_reference**
- typedef _Ht::difference_type **difference_type**
- typedef _Ht::hasher **hasher**
- typedef _Ht::const_iterator **iterator**
- typedef _Ht::key_equal **key_equal**
- typedef _Ht::key_type **key_type**
- typedef _Alloc::pointer **pointer**
- typedef _Alloc::reference **reference**
- typedef _Ht::size_type **size_type**
- typedef _Ht::value_type **value_type**

Public Member Functions

- **hash_multiset** (size_type __n)
- **hash_multiset** (size_type __n, const hasher &__hf)
- **hash_multiset** (size_type __n, const hasher &__hf, const key_equal &__eq, const allocator_type &__a=allocator_type())
- template<class _InputIterator> **hash_multiset** (_InputIterator __f, _InputIterator __l)
- template<class _InputIterator> **hash_multiset** (_InputIterator __f, _InputIterator __l, size_type __n)
- template<class _InputIterator> **hash_multiset** (_InputIterator __f, _InputIterator __l, size_type __n, const hasher &__hf)

- `template<class _InputIterator >`
`hash_multiset` (`_InputIterator __f, _InputIterator __l, size_type __n, const hasher &__hf, const key_equal &__eq,`
`const allocator_type &__a=allocator_type()`)
- iterator `begin` () const
- `size_type bucket_count` () const
- void `clear` ()
- `size_type count` (const `key_type &__key`) const
- `size_type elems_in_bucket` (`size_type __n`) const
- bool `empty` () const
- iterator `end` () const
- `pair< iterator, iterator > equal_range` (const `key_type &__key`) const
- `size_type erase` (const `key_type &__key`)
- void `erase` (iterator `__it`)
- void `erase` (iterator `__f, iterator __l`)
- iterator `find` (const `key_type &__key`) const
- `allocator_type get_allocator` () const
- hasher `hash_funct` () const
- iterator `insert` (const `value_type &__obj`)
- `template<class _InputIterator >`
void `insert` (`_InputIterator __f, _InputIterator __l`)
- iterator `insert_noresize` (const `value_type &__obj`)
- `key_equal key_eq` () const
- `size_type max_bucket_count` () const
- `size_type max_size` () const
- void `resize` (`size_type __hint`)
- `size_type size` () const
- void `swap` (`hash_multiset &hs`)

Friends

- `template<class _Val, class _HF, class _EqK, class _Al >`
bool `operator==` (const `hash_multiset< _Val, _HF, _EqK, _Al > &`, const `hash_multiset< _Val, _HF, _EqK, _Al >` &)

5.47.1 Detailed Description

```
template<class _Value, class _HashFcn = hash<_Value>, class _EqualKey = equal_to<_Value>, class _Alloc = allocator<_Value>>
class __gnu_cxx::hash_multiset< _Value, _HashFcn, _EqualKey, _Alloc >
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation_style.html

Definition at line 285 of file `hash_set`.

The documentation for this class was generated from the following file:

- [hash_set](#)

5.48 `__gnu_cxx::hash_set<_Value, _HashFcn, _EqualKey, _Alloc >` Class Template Reference

Public Types

- `typedef _Ht::allocator_type allocator_type`
- `typedef _Ht::const_iterator const_iterator`
- `typedef _Alloc::const_pointer const_pointer`
- `typedef _Alloc::const_reference const_reference`
- `typedef _Ht::difference_type difference_type`
- `typedef _Ht::hasher hasher`
- `typedef _Ht::const_iterator iterator`
- `typedef _Ht::key_equal key_equal`
- `typedef _Ht::key_type key_type`
- `typedef _Alloc::pointer pointer`
- `typedef _Alloc::reference reference`
- `typedef _Ht::size_type size_type`
- `typedef _Ht::value_type value_type`

Public Member Functions

- `hash_set (size_type __n)`
- `hash_set (size_type __n, const hasher &__hf)`
- `hash_set (size_type __n, const hasher &__hf, const key_equal &__eq, const allocator_type &__a=allocator_type())`
- `template<class _InputIterator >
hash_set (_InputIterator __f, _InputIterator __l)`
- `template<class _InputIterator >
hash_set (_InputIterator __f, _InputIterator __l, size_type __n)`
- `template<class _InputIterator >
hash_set (_InputIterator __f, _InputIterator __l, size_type __n, const hasher &__hf)`
- `template<class _InputIterator >
hash_set (_InputIterator __f, _InputIterator __l, size_type __n, const hasher &__hf, const key_equal &__eq,
const allocator_type &__a=allocator_type())`
- `iterator begin () const`
- `size_type bucket_count () const`
- `void clear ()`
- `size_type count (const key_type &__key) const`
- `size_type elems_in_bucket (size_type __n) const`
- `bool empty () const`
- `iterator end () const`
- `pair< iterator, iterator > equal_range (const key_type &__key) const`
- `size_type erase (const key_type &__key)`
- `void erase (iterator __it)`
- `void erase (iterator __f, iterator __l)`
- `iterator find (const key_type &__key) const`
- `allocator_type get_allocator () const`
- `hasher hash_funct () const`
- `pair< iterator, bool > insert (const value_type &__obj)`
- `template<class _InputIterator >
void insert (_InputIterator __f, _InputIterator __l)`

- [pair](#)< iterator, bool > **insert_noresize** (const value_type &__obj)
- key_equal **key_eq** () const
- size_type **max_bucket_count** () const
- size_type **max_size** () const
- void **resize** (size_type __hint)
- size_type **size** () const
- void **swap** ([hash_set](#) &__hs)

Friends

- template<class _Val, class _HF, class _EqK, class _Al >
bool **operator==** (const [hash_set](#)< _Val, _HF, _EqK, _Al > &, const [hash_set](#)< _Val, _HF, _EqK, _Al > &)

5.48.1 Detailed Description

```
template<class _Value, class _HashFcn = hash<_Value>, class _EqualKey = equal_to<_Value>, class _Alloc = allocator<_↵
_Value>>
class __gnu_cxx::hash_set< _Value, _HashFcn, _EqualKey, _Alloc >
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation↵_style.html

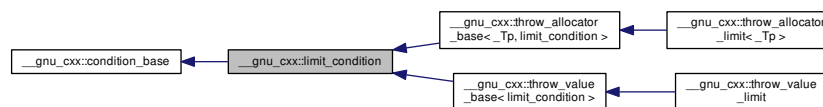
Definition at line 84 of file hash_set.

The documentation for this class was generated from the following file:

- [hash_set](#)

5.49 __gnu_cxx::limit_condition Struct Reference

Inheritance diagram for __gnu_cxx::limit_condition:



Classes

- struct [always_adjustor](#)
- struct [limit_adjustor](#)
- struct [never_adjustor](#)

Static Public Member Functions

- static size_t & **count** ()
- static size_t & **limit** ()
- static void **set_limit** (const size_t __l)
- static void **throw_conditionally** ()

5.49.1 Detailed Description

Base class for incremental control and throw.

Definition at line 412 of file `throw_allocator.h`.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

5.50 __gnu_cxx::limit_condition::always_adjustor Struct Reference

Inherits `__gnu_cxx::limit_condition::adjustor_base`.

5.50.1 Detailed Description

Always enter the condition.

Definition at line 436 of file `throw_allocator.h`.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

5.51 __gnu_cxx::limit_condition::limit_adjustor Struct Reference

Inherits `__gnu_cxx::limit_condition::adjustor_base`.

Public Member Functions

- **limit_adjustor** (const size_t __l)

5.51.1 Detailed Description

Enter the nth condition.

Definition at line 442 of file `throw_allocator.h`.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

5.52 `__gnu_cxx::limit_condition::never_adjustor` Struct Reference

Inherits `__gnu_cxx::limit_condition::adjustor_base`.

5.52.1 Detailed Description

Never enter the condition.

Definition at line 430 of file `throw_allocator.h`.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

5.53 `__gnu_cxx::malloc_allocator<_Tp>` Class Template Reference

Public Types

- typedef const _Tp * **const_pointer**
- typedef const _Tp & **const_reference**
- typedef ptrdiff_t **difference_type**
- typedef _Tp * **pointer**
- typedef [std::true_type](#) **propagate_on_container_move_assignment**
- typedef _Tp & **reference**
- typedef size_t **size_type**
- typedef _Tp **value_type**

Public Member Functions

- **malloc_allocator** (const [malloc_allocator](#) &) noexcept
- template<typename _Tp1 >
 malloc_allocator (const [malloc_allocator](#)< _Tp1 > &) noexcept
- pointer **address** (reference __x) const noexcept
- const_pointer **address** (const_reference __x) const noexcept
- pointer **allocate** (size_type __n, const void *=0)
- template<typename _Up, typename... _Args>
 void **construct** (_Up *__p, _Args &&... __args)
- void **deallocate** (pointer __p, size_type)
- template<typename _Up >
 void **destroy** (_Up *__p)
- size_type **max_size** () const noexcept

5.53.1 Detailed Description

```
template<typename _Tp>
class __gnu_cxx::malloc_allocator< _Tp >
```

An allocator that uses malloc.

This is precisely the allocator defined in the C++ Standard.

- all allocation calls malloc
- all deallocation calls free

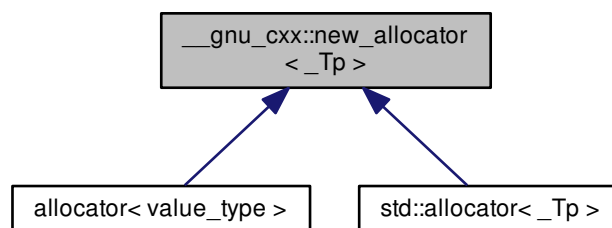
Definition at line 56 of file malloc_allocator.h.

The documentation for this class was generated from the following file:

- [malloc_allocator.h](#)

5.54 __gnu_cxx::new_allocator< _Tp > Class Template Reference

Inheritance diagram for __gnu_cxx::new_allocator< _Tp >:



Public Types

- typedef const _Tp * **const_pointer**
- typedef const _Tp & **const_reference**
- typedef ptrdiff_t **difference_type**
- typedef _Tp * **pointer**
- typedef [std::true_type](#) **propagate_on_container_move_assignment**
- typedef _Tp & **reference**
- typedef size_t **size_type**
- typedef _Tp **value_type**

Public Member Functions

- **new_allocator** (const [new_allocator](#) &) noexcept
- template<typename _Tp1 >
 new_allocator (const [new_allocator](#)<_Tp1> &) noexcept
- pointer **address** (reference __x) const noexcept
- const_pointer **address** (const_reference __x) const noexcept
- pointer **allocate** (size_type __n, const void * = 0)
- template<typename _Up, typename... _Args>
 void **construct** (_Up * __p, _Args &&... __args)
- void **deallocate** (pointer __p, size_type)
- template<typename _Up >
 void **destroy** (_Up * __p)
- size_type **max_size** () const noexcept

5.54.1 Detailed Description

```
template<typename _Tp>
class __gnu_cxx::new_allocator<_Tp>
```

An allocator that uses global new, as per [20.4].

This is precisely the allocator defined in the C++ Standard.

- all allocation calls operator new
- all deallocation calls operator delete

Template Parameters

<code>_Tp</code>	Type of allocated object.
------------------	---------------------------

Definition at line 58 of file `new_allocator.h`.

The documentation for this class was generated from the following file:

- [new_allocator.h](#)

5.55 `__gnu_cxx::project1st<_Arg1, _Arg2>` Struct Template Reference

Inherits `__gnu_cxx::_Project1st<_Arg1, _Arg2>`.

Public Types

- typedef `_Arg1` [first_argument_type](#)
- typedef `_Result` [result_type](#)
- typedef `_Arg2` [second_argument_type](#)

Public Member Functions

- `_Arg1 operator()` (`const _Arg1 &__x`, `const _Arg2 &`) `const`

5.55.1 Detailed Description

```
template<class _Arg1, class _Arg2>
struct __gnu_cxx::project1st<_Arg1, _Arg2>
```

An [SGI extension](#) .

Definition at line 237 of file `ext/functional`.

5.55.2 Member Typedef Documentation

5.55.2.1 `template<typename _Arg1, typename _Arg2, typename _Result> typedef _Arg1 std::binary_function<_Arg1, _Arg2, _Result>::first_argument_type` [\[inherited\]](#)

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.55.2.2 `template<typename _Arg1, typename _Arg2, typename _Result> typedef _Result std::binary_function<_Arg1, _Arg2, _Result>::result_type` [\[inherited\]](#)

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.55.2.3 `template<typename _Arg1, typename _Arg2, typename _Result> typedef _Arg2 std::binary_function<_Arg1, _Arg2, _Result>::second_argument_type` [\[inherited\]](#)

`second_argument_type` is the type of the second argument

Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [ext/functional](#)

5.56 `__gnu_cxx::project2nd<_Arg1, _Arg2>` Struct Template Reference

Inherits `__gnu_cxx::_Project2nd<_Arg1, _Arg2>`.

Public Types

- typedef `_Arg1` [first_argument_type](#)
- typedef `_Result` [result_type](#)
- typedef `_Arg2` [second_argument_type](#)

Public Member Functions

- `_Arg2 operator()` (const `_Arg1` &, const `_Arg2` &__y) const

5.56.1 Detailed Description

```
template<class _Arg1, class _Arg2>
struct __gnu_cxx::project2nd<_Arg1, _Arg2>
```

An [SGI extension](#) .

Definition at line 241 of file `ext/functional`.

5.56.2 Member Typedef Documentation

5.56.2.1 `template<typename _Arg1, typename _Arg2, typename _Result> typedef _Arg1 std::binary_function<_Arg1, _Arg2, _Result>::first_argument_type` [\[inherited\]](#)

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.56.2.2 `template<typename _Arg1, typename _Arg2, typename _Result> typedef _Result std::binary_function<_Arg1, _Arg2, _Result>::result_type` [\[inherited\]](#)

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.56.2.3 `template<typename _Arg1, typename _Arg2, typename _Result> typedef _Arg2 std::binary_function<_Arg1, _Arg2, _Result>::second_argument_type` [\[inherited\]](#)

`second_argument_type` is the type of the second argument

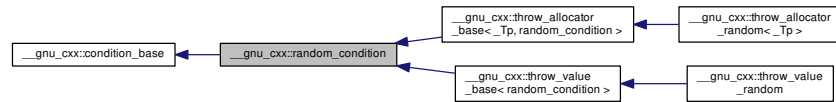
Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [ext/functional](#)

5.57 `__gnu_cxx::random_condition` Struct Reference

Inheritance diagram for `__gnu_cxx::random_condition`:



Classes

- struct [always_adjustor](#)
- struct [group_adjustor](#)
- struct [never_adjustor](#)

Public Member Functions

- void **seed** (unsigned long __s)

Static Public Member Functions

- static void **set_probability** (double __p)
- static void **throw_conditionally** ()

5.57.1 Detailed Description

Base class for random probability control and throw.

Definition at line 484 of file `throw_allocator.h`.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

5.58 `__gnu_cxx::random_condition::always_adjustor` Struct Reference

Inherits `__gnu_cxx::random_condition::adjustor_base`.

5.58.1 Detailed Description

Always enter the condition.

Definition at line 517 of file `throw_allocator.h`.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

5.59 `__gnu_cxx::random_condition::group_adjustor` Struct Reference

Inherits `__gnu_cxx::random_condition::adjustor_base`.

Public Member Functions

- **`group_adjustor`** (`size_t` size)

5.59.1 Detailed Description

Group condition.

Definition at line 502 of file `throw_allocator.h`.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

5.60 `__gnu_cxx::random_condition::never_adjustor` Struct Reference

Inherits `__gnu_cxx::random_condition::adjustor_base`.

5.60.1 Detailed Description

Never enter the condition.

Definition at line 511 of file `throw_allocator.h`.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

5.61 `__gnu_cxx::rb_tree<_Key, _Value, _KeyOfValue, _Compare, _Alloc >` Struct Template Reference

Inherits `std::Rb_tree<_Key, _Val, _KeyOfValue, _Compare, _Alloc >`.

Public Types

- `typedef _Rb_tree<_Key, _Value, _KeyOfValue, _Compare, _Alloc > _Base`
- `typedef _Base::allocator_type allocator_type`
- `typedef _Rb_tree_const_iterator<value_type> const_iterator`
- `typedef const value_type * const_pointer`
- `typedef const value_type & const_reference`
- `typedef std::reverse_iterator<const_iterator> const_reverse_iterator`
- `typedef ptrdiff_t difference_type`
- `typedef _Rb_tree_iterator<value_type> iterator`
- `typedef _Key key_type`
- `typedef value_type * pointer`
- `typedef value_type & reference`
- `typedef std::reverse_iterator<iterator> reverse_iterator`
- `typedef size_t size_type`
- `typedef _Val value_type`

Public Member Functions

- `rb_tree (const _Compare &__comp=_Compare(), const allocator_type &__a=allocator_type())`
- `bool __rb_verify () const`
- `template<typename _Iterator >
void _M_assign_equal (_Iterator, _Iterator)`
- `template<typename _Iterator >
void _M_assign_unique (_Iterator, _Iterator)`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
size_type _M_count_tr (const _Kt &__k) const`
- `template<typename... _Args>
iterator _M_emplace_equal (_Args &&...__args)`
- `template<typename... _Args>
iterator _M_emplace_hint_equal (const_iterator __pos, _Args &&...__args)`
- `template<typename... _Args>
iterator _M_emplace_hint_unique (const_iterator __pos, _Args &&...__args)`
- `template<typename... _Args>
pair<iterator, bool> _M_emplace_unique (_Args &&...__args)`
- `template<typename... _Args>
pair<typename _Rb_tree<_Key, _Val, _KeyOfValue, _Compare, _Alloc>::iterator, bool> _M_emplace_↵
unique (_Args &&...__args)`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
pair<iterator, iterator> _M_equal_range_tr (const _Kt &__k)`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
pair<const_iterator, const_iterator> _M_equal_range_tr (const _Kt &__k) const`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
iterator _M_find_tr (const _Kt &__k)`

- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`const_iterator` **`_M_find_tr`** (`const _Kt &__k`) `const`
- `pair<_Base_ptr, _Base_ptr>` **`_M_get_insert_equal_pos`** (`const key_type &__k`)
- `pair<_Base_ptr, _Base_ptr>` **`_M_get_insert_hint_equal_pos`** (`const_iterator __pos, const key_type &__k`)
- `pair<_Base_ptr, _Base_ptr>` **`_M_get_insert_hint_unique_pos`** (`const_iterator __pos, const key_type &__k`)
- `pair<_Base_ptr, _Base_ptr>` **`_M_get_insert_unique_pos`** (`const key_type &__k`)
- `_Node_allocator &` **`_M_get_Node_allocator`** () `noexcept`
- `const _Node_allocator &` **`_M_get_Node_allocator`** () `const noexcept`
- `template<typename _Arg>`
`iterator` **`_M_insert_equal`** (`_Arg &&__x`)
- `template<typename _InputIterator>`
`void` **`_M_insert_equal`** (`_InputIterator __first, _InputIterator __last`)
- `template<class _II>`
`void` **`_M_insert_equal`** (`_II __first, _II __last`)
- `template<typename _Arg, typename _NodeGen>`
`iterator` **`_M_insert_equal_`** (`const_iterator __pos, _Arg &&__x, _NodeGen &`)
- `template<typename _Arg>`
`iterator` **`_M_insert_equal_`** (`const_iterator __pos, _Arg &&__x`)
- `template<typename _Arg>`
`pair<iterator, bool>` **`_M_insert_unique`** (`_Arg &&__x`)
- `template<typename _InputIterator>`
`void` **`_M_insert_unique`** (`_InputIterator __first, _InputIterator __last`)
- `template<typename _Arg>`
`pair<typename _Rb_tree<_Key, _Val, _KeyOfValue, _Compare, _Alloc>::iterator, bool>` **`_M_insert_unique`**
(`_Arg &&__v`)
- `template<class _II>`
`void` **`_M_insert_unique`** (`_II __first, _II __last`)
- `template<typename _Arg, typename _NodeGen>`
`iterator` **`_M_insert_unique_`** (`const_iterator __pos, _Arg &&__x, _NodeGen &`)
- `template<typename _Arg>`
`iterator` **`_M_insert_unique_`** (`const_iterator __pos, _Arg &&__x`)
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`iterator` **`_M_lower_bound_tr`** (`const _Kt &__k`)
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`const_iterator` **`_M_lower_bound_tr`** (`const _Kt &__k`) `const`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`iterator` **`_M_upper_bound_tr`** (`const _Kt &__k`)
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`const_iterator` **`_M_upper_bound_tr`** (`const _Kt &__k`) `const`
- `iterator` **`begin`** () `noexcept`
- `const_iterator` **`begin`** () `const noexcept`
- `void` **`clear`** () `noexcept`
- `size_type` **`count`** (`const key_type &__k`) `const`
- `bool` **`empty`** () `const noexcept`
- `iterator` **`end`** () `noexcept`
- `const_iterator` **`end`** () `const noexcept`
- `pair<iterator, iterator>` **`equal_range`** (`const key_type &__k`)
- `pair<const_iterator, const_iterator>` **`equal_range`** (`const key_type &__k`) `const`
- `_GLIBCXX_ABI_TAG_CXX11 iterator` **`erase`** (`const_iterator __position`)
- `_GLIBCXX_ABI_TAG_CXX11 iterator` **`erase`** (`iterator __position`)
- `size_type` **`erase`** (`const key_type &__x`)
- `_GLIBCXX_ABI_TAG_CXX11 iterator` **`erase`** (`const_iterator __first, const_iterator __last`)

- void **erase** (const key_type *__first, const key_type *__last)
- iterator **find** (const key_type &__k)
- const_iterator **find** (const key_type &__k) const
- allocator_type **get_allocator** () const noexcept
- _Compare **key_comp** () const
- iterator **lower_bound** (const key_type &__k)
- const_iterator **lower_bound** (const key_type &__k) const
- size_type **max_size** () const noexcept
- [reverse_iterator](#) **rbegin** () noexcept
- [const_reverse_iterator](#) **rbegin** () const noexcept
- [reverse_iterator](#) **rend** () noexcept
- [const_reverse_iterator](#) **rend** () const noexcept
- size_type **size** () const noexcept
- void **swap** (_Rb_tree &__t) noexcept(/*conditional */)
- iterator **upper_bound** (const key_type &__k)
- const_iterator **upper_bound** (const key_type &__k) const

Protected Types

- typedef _Rb_tree_node_base * **_Base_ptr**
- typedef const _Rb_tree_node_base * **_Const_Base_ptr**
- typedef const _Rb_tree_node< _Val > * **_Const_Link_type**
- typedef _Rb_tree_node< _Val > * **_Link_type**

Protected Member Functions

- _Link_type **_M_begin** () noexcept
- _Const_Link_type **_M_begin** () const noexcept
- template<typename _NodeGen >
_Link_type **_M_clone_node** (_Const_Link_type __x, _NodeGen &__node_gen)
- template<typename... _Args>
void **_M_construct_node** (_Link_type __node, _Args &&... __args)
- template<typename... _Args>
_Link_type **_M_create_node** (_Args &&... __args)
- void **_M_destroy_node** (_Link_type __p) noexcept
- void **_M_drop_node** (_Link_type __p) noexcept
- _Base_ptr **_M_end** () noexcept
- _Const_Base_ptr **_M_end** () const noexcept
- _Link_type **_M_get_node** ()
- _Base_ptr & **_M_leftmost** () noexcept
- _Const_Base_ptr **_M_leftmost** () const noexcept
- void **_M_put_node** (_Link_type __p) noexcept
- _Base_ptr & **_M_rightmost** () noexcept
- _Const_Base_ptr **_M_rightmost** () const noexcept
- _Base_ptr & **_M_root** () noexcept
- _Const_Base_ptr **_M_root** () const noexcept

Static Protected Member Functions

- static const _Key & **_S_key** (_Const_Link_type __x)
- static const _Key & **_S_key** (_Const_Base_ptr __x)
- static _Link_type **_S_left** (_Base_ptr __x) noexcept
- static _Const_Link_type **_S_left** (_Const_Base_ptr __x) noexcept
- static _Base_ptr **_S_maximum** (_Base_ptr __x) noexcept
- static _Const_Base_ptr **_S_maximum** (_Const_Base_ptr __x) noexcept
- static _Base_ptr **_S_minimum** (_Base_ptr __x) noexcept
- static _Const_Base_ptr **_S_minimum** (_Const_Base_ptr __x) noexcept
- static _Link_type **_S_right** (_Base_ptr __x) noexcept
- static _Const_Link_type **_S_right** (_Const_Base_ptr __x) noexcept
- static const_reference **_S_value** (_Const_Link_type __x)
- static const_reference **_S_value** (_Const_Base_ptr __x)

Protected Attributes

- _Rb_tree_impl< _Compare > **_M_impl**

5.61.1 Detailed Description

```
template<class _Key, class _Value, class _KeyOfValue, class _Compare, class _Alloc = allocator<_Value>>
struct __gnu_cxx::rb_tree< _Key, _Value, _KeyOfValue, _Compare, _Alloc >
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation_style.html

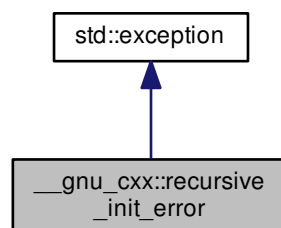
Definition at line 80 of file rb_tree.

The documentation for this struct was generated from the following file:

- [rb_tree](#)

5.62 __gnu_cxx::recursive_init_error Class Reference

Inheritance diagram for __gnu_cxx::recursive_init_error:



Public Member Functions

- virtual const char * [what](#) () const _GLIBCXX_TXN_SAFE_DYN noexcept

5.62.1 Detailed Description

Exception thrown by `__cxa_guard_acquire`.

6.7[stmt.dcl]/4: If control re-enters the declaration (recursively) while the object is being initialized, the behavior is undefined.

Since we already have a library function to handle locking, we might as well check for this situation and throw an exception. We use the second byte of the guard variable to remember that we're in the middle of an initialization.

Definition at line 700 of file `cxxabi.h`.

5.62.2 Member Function Documentation

5.62.2.1 `virtual const char* std::exception::what () const` `[virtual]`, `[noexcept]`, `[inherited]`

Returns a C-style character string describing the general cause of the current error.

Reimplemented in `std::bad_function_call`, `std::ios_base::failure`, `std::runtime_error`, `std::bad_typeid`, `std::bad_cast`, `std::logic_error`, `std::future_error`, `std::bad_exception`, `std::bad_weak_ptr`, `std::experimental::fundamentals_v1::bad_↵ any_cast`, and `std::bad_alloc`.

The documentation for this class was generated from the following file:

- [cxxabi.h](#)

5.63 `__gnu_cxx::rope<_CharT, _Alloc>` Class Template Reference

Inherits `__gnu_cxx::Rope_base<_CharT, _Alloc>`.

Public Types

- typedef `_Rope_RopeConcatenation<_CharT, _Alloc>` `__C`
- typedef `_Rope_RopeFunction<_CharT, _Alloc>` `__F`
- typedef `_Rope_RopeLeaf<_CharT, _Alloc>` `__L`
- typedef `_Rope_RopeSubstring<_CharT, _Alloc>` `__S`
- typedef `_Alloc::template rebind<__C>::other` `_CAlloc`
- typedef `_Alloc::template rebind<_CharT>::other` `_DataAlloc`
- typedef `_Alloc::template rebind<__F>::other` `_FAlloc`
- typedef `_Alloc::template rebind<__L>::other` `_LAlloc`
- typedef `_Alloc::template rebind<__S>::other` `_SAlloc`
- typedef `_Rope_const_iterator<_CharT, _Alloc>` `const_iterator`
- typedef `const _CharT *` `const_pointer`
- typedef `_CharT` `const_reference`
- typedef `std::reverse_iterator<const_iterator>` `const_reverse_iterator`
- typedef `ptrdiff_t` `difference_type`
- typedef `_Rope_iterator<_CharT, _Alloc>` `iterator`
- typedef `_Rope_char_ptr_proxy<_CharT, _Alloc>` `pointer`
- typedef `_Rope_char_ref_proxy<_CharT, _Alloc>` `reference`
- typedef `std::reverse_iterator<iterator>` `reverse_iterator`
- typedef `size_t` `size_type`
- typedef `_CharT` `value_type`

Public Member Functions

- **rope** (const `_CharT` *`__s`, const `allocator_type` &`__a`=`allocator_type`())
- **rope** (const `_CharT` *`__s`, `size_t` `__len`, const `allocator_type` &`__a`=`allocator_type`())
- **rope** (const `_CharT` *`__s`, const `_CharT` *`__e`, const `allocator_type` &`__a`=`allocator_type`())
- **rope** (const const_iterator &`__s`, const const_iterator &`__e`, const `allocator_type` &`__a`=`allocator_type`())
- **rope** (const iterator &`__s`, const iterator &`__e`, const `allocator_type` &`__a`=`allocator_type`())
- **rope** (`_CharT` `__c`, const `allocator_type` &`__a`=`allocator_type`())
- **rope** (`size_t` `__n`, `_CharT` `__c`, const `allocator_type` &`__a`=`allocator_type`())
- **rope** (const `allocator_type` &`__a`=`allocator_type`())
- **rope** (char_producer< `_CharT` > *`__fn`, `size_t` `__len`, bool `__delete_fn`, const `allocator_type` &`__a`=`allocator_type`())
- **rope** (const `rope` &`__x`, const `allocator_type` &`__a`=`allocator_type`())
- `allocator_type` & **M_get_allocator** ()
- const `allocator_type` & **M_get_allocator** () const
- `rope` & **append** (const `_CharT` *`__iter`, `size_t` `__n`)
- `rope` & **append** (const `_CharT` *`__c_string`)
- `rope` & **append** (const `_CharT` *`__s`, const `_CharT` *`__e`)
- `rope` & **append** (const_iterator `__s`, const_iterator `__e`)
- `rope` & **append** (`_CharT` `__c`)
- `rope` & **append** ()
- `rope` & **append** (const `rope` &`__y`)
- `rope` & **append** (`size_t` `__n`, `_CharT` `__c`)
- void **apply_to_pieces** (`size_t` `__begin`, `size_t` `__end`, `Rope_char_consumer`< `_CharT` > &`__c`) const
- `_CharT` **at** (`size_type` `__pos`) const
- `_CharT` **back** () const
- void **balance** ()
- const_iterator **begin** () const
- const_iterator **begin** ()
- const `_CharT` * **c_str** () const
- void **clear** ()
- int **compare** (const `rope` &`__y`) const
- const_iterator **const_begin** () const
- const_iterator **const_end** () const
- const_reverse_iterator **const_rbegin** () const
- const_reverse_iterator **const_rend** () const
- void **copy** (`_CharT` *`__buffer`) const
- `size_type` **copy** (`size_type` `__pos`, `size_type` `__n`, `_CharT` *`__buffer`) const
- void **delete_c_str** ()
- void **dump** ()
- bool **empty** () const
- const_iterator **end** () const
- const_iterator **end** ()
- void **erase** (`size_t` `__p`, `size_t` `__n`)
- void **erase** (`size_t` `__p`)
- iterator **erase** (const iterator &`__p`, const iterator &`__q`)
- iterator **erase** (const iterator &`__p`)
- `size_type` **find** (`_CharT` `__c`, `size_type` `__pos`=0) const
- `size_type` **find** (const `_CharT` *`__s`, `size_type` `__pos`=0) const
- `_CharT` **front** () const
- `allocator_type` **get_allocator** () const

- void **insert** (size_t __p, const rope &__r)
- void **insert** (size_t __p, size_t __n, _CharT __c)
- void **insert** (size_t __p, const _CharT *__i, size_t __n)
- void **insert** (size_t __p, const _CharT *__c_string)
- void **insert** (size_t __p, _CharT __c)
- void **insert** (size_t __p)
- void **insert** (size_t __p, const _CharT *__i, const _CharT *__j)
- void **insert** (size_t __p, const const_iterator &__i, const const_iterator &__j)
- void **insert** (size_t __p, const iterator &__i, const iterator &__j)
- iterator **insert** (const iterator &__p, const rope &__r)
- iterator **insert** (const iterator &__p, size_t __n, _CharT __c)
- iterator **insert** (const iterator &__p, _CharT __c)
- iterator **insert** (const iterator &__p)
- iterator **insert** (const iterator &__p, const _CharT *c_string)
- iterator **insert** (const iterator &__p, const _CharT *__i, size_t __n)
- iterator **insert** (const iterator &__p, const _CharT *__i, const _CharT *__j)
- iterator **insert** (const iterator &__p, const const_iterator &__i, const const_iterator &__j)
- iterator **insert** (const iterator &__p, const iterator &__i, const iterator &__j)
- size_type **length** () const
- size_type **max_size** () const
- iterator **mutable_begin** ()
- iterator **mutable_end** ()
- [reverse_iterator](#) **mutable_rbegin** ()
- reference **mutable_reference_at** (size_type __pos)
- [reverse_iterator](#) **mutable_rend** ()
- rope & **operator=** (const rope &__x)
- _CharT **operator[]** (size_type __pos) const
- void **pop_back** ()
- void **pop_front** ()
- void **push_back** (_CharT __x)
- void **push_front** (_CharT __x)
- [const_reverse_iterator](#) **rbegin** () const
- [const_reverse_iterator](#) **rbegin** ()
- [const_reverse_iterator](#) **rend** () const
- [const_reverse_iterator](#) **rend** ()
- void **replace** (size_t __p, size_t __n, const rope &__r)
- void **replace** (size_t __p, size_t __n, const _CharT *__i, size_t __i_len)
- void **replace** (size_t __p, size_t __n, _CharT __c)
- void **replace** (size_t __p, size_t __n, const _CharT *__c_string)
- void **replace** (size_t __p, size_t __n, const _CharT *__i, const _CharT *__j)
- void **replace** (size_t __p, size_t __n, const const_iterator &__i, const const_iterator &__j)
- void **replace** (size_t __p, size_t __n, const iterator &__i, const iterator &__j)
- void **replace** (size_t __p, _CharT __c)
- void **replace** (size_t __p, const rope &__r)
- void **replace** (size_t __p, const _CharT *__i, size_t __i_len)
- void **replace** (size_t __p, const _CharT *__c_string)
- void **replace** (size_t __p, const _CharT *__i, const _CharT *__j)
- void **replace** (size_t __p, const const_iterator &__i, const const_iterator &__j)
- void **replace** (size_t __p, const iterator &__i, const iterator &__j)
- void **replace** (const iterator &__p, const iterator &__q, const rope &__r)
- void **replace** (const iterator &__p, const iterator &__q, _CharT __c)

- void **replace** (const iterator &__p, const iterator &__q, const _CharT *__c_string)
- void **replace** (const iterator &__p, const iterator &__q, const _CharT *__i, size_t __n)
- void **replace** (const iterator &__p, const iterator &__q, const _CharT *__i, const _CharT *__j)
- void **replace** (const iterator &__p, const iterator &__q, const const_iterator &__i, const const_iterator &__j)
- void **replace** (const iterator &__p, const iterator &__q, const iterator &__i, const iterator &__j)
- void **replace** (const iterator &__p, const [rope](#) &__r)
- void **replace** (const iterator &__p, _CharT __c)
- void **replace** (const iterator &__p, const _CharT *__c_string)
- void **replace** (const iterator &__p, const _CharT *__i, size_t __n)
- void **replace** (const iterator &__p, const _CharT *__i, const _CharT *__j)
- void **replace** (const iterator &__p, const_iterator __i, const_iterator __j)
- void **replace** (const iterator &__p, iterator __i, iterator __j)
- const _CharT * **replace_with_c_str** ()
- size_type **size** () const
- [rope](#) **substr** (size_t __start, size_t __len=1) const
- [rope](#) **substr** (iterator __start, iterator __end) const
- [rope](#) **substr** (iterator __start) const
- [rope](#) **substr** (const_iterator __start, const_iterator __end) const
- [rope](#)<_CharT, _Alloc> **substr** (const_iterator __start)
- void **swap** ([rope](#) &__b)

Static Public Member Functions

- static __C * **_C_allocate** (size_t __n)
- static void **_C_deallocate** (__C *__p, size_t __n)
- static _CharT * **_Data_allocate** (size_t __n)
- static void **_Data_deallocate** (_CharT *__p, size_t __n)
- static __F * **_F_allocate** (size_t __n)
- static void **_F_deallocate** (__F *__p, size_t __n)
- static __L * **_L_allocate** (size_t __n)
- static void **_L_deallocate** (__L *__p, size_t __n)
- static __S * **_S_allocate** (size_t __n)
- static void **_S_deallocate** (__S *__p, size_t __n)

Public Attributes

- _RopeRep * **_M_tree_ptr**

Static Public Attributes

- static const size_type **npos**

Protected Types

- enum { **_S_copy_max** }
- typedef _Rope_base< _CharT, _Alloc > **_Base**
- typedef _CharT * **_Cstrptr**
- typedef _Rope_RopeConcatenation< _CharT, _Alloc > **_RopeConcatenation**
- typedef _Rope_RopeFunction< _CharT, _Alloc > **_RopeFunction**
- typedef _Rope_RopeLeaf< _CharT, _Alloc > **_RopeLeaf**
- typedef _Rope_RopeRep< _CharT, _Alloc > **_RopeRep**
- typedef _Rope_RopeSubstring< _CharT, _Alloc > **_RopeSubstring**
- typedef _Rope_self_destruct_ptr< _CharT, _Alloc > **_Self_destruct_ptr**
- typedef _Base::allocator_type **allocator_type**

Static Protected Member Functions

- static size_t **_S_allocated_capacity** (size_t __n)
- static bool **_S_apply_to_pieces** (_Rope_char_consumer< _CharT > &__c, const _RopeRep * __r, size_t __↵begin, size_t __end)
- static _RopeRep * **_S_concat** (_RopeRep * __left, _RopeRep * __right)
- static _RopeRep * **_S_concat_char_iter** (_RopeRep * __r, const _CharT * __iter, size_t __slen)
- static _RopeRep * **_S_destr_concat_char_iter** (_RopeRep * __r, const _CharT * __iter, size_t __slen)
- static _RopeLeaf * **_S_destr_leaf_concat_char_iter** (_RopeLeaf * __r, const _CharT * __iter, size_t __slen)
- static _CharT **_S_fetch** (_RopeRep * __r, size_type __pos)
- static _CharT * **_S_fetch_ptr** (_RopeRep * __r, size_type __pos)
- static bool **_S_is0** (_CharT __c)
- static _RopeLeaf * **_S_leaf_concat_char_iter** (_RopeLeaf * __r, const _CharT * __iter, size_t __slen)
- static _RopeConcatenation * **_S_new_RopeConcatenation** (_RopeRep * __left, _RopeRep * __right, allocator↵_type & __a)
- static _RopeFunction * **_S_new_RopeFunction** (char_producer< _CharT > * __f, size_t __size, bool __↵d, allocator_type & __a)
- static _RopeLeaf * **_S_new_RopeLeaf** (_CharT * __s, size_t __size, allocator_type & __a)
- static _RopeSubstring * **_S_new_RopeSubstring** (_Rope_RopeRep< _CharT, _Alloc > * __b, size_t __s, size↵_t __l, allocator_type & __a)
- static void **_S_ref** (_RopeRep * __t)
- static _RopeLeaf * **_S_RopeLeaf_from_unowned_char_ptr** (const _CharT * __s, size_t __size, allocator_type↵ & __a)
- static size_t **_S_rounded_up_size** (size_t __n)
- static _RopeRep * **_S_substring** (_RopeRep * __base, size_t __start, size_t __endp1)
- static _RopeRep * **_S_tree_concat** (_RopeRep * __left, _RopeRep * __right)
- static void **_S_unref** (_RopeRep * __t)
- static _RopeRep * **replace** (_RopeRep * __old, size_t __pos1, size_t __pos2, _RopeRep * __r)

Static Protected Attributes

- static _CharT **_S_empty_c_str** [1]

Friends

- class `_Rope_char_ptr_proxy<_CharT, _Alloc>`
- class `_Rope_char_ref_proxy<_CharT, _Alloc>`
- class `_Rope_const_iterator<_CharT, _Alloc>`
- class `_Rope_iterator<_CharT, _Alloc>`
- class `_Rope_iterator_base<_CharT, _Alloc>`
- struct `_Rope_RopeRep<_CharT, _Alloc>`
- struct `_Rope_RopeSubstring<_CharT, _Alloc>`
- template<class `_CharT2`, class `_Alloc2`>
`rope<_CharT2, _Alloc2> operator+ (const rope<_CharT2, _Alloc2> &__left, const rope<_CharT2, _Alloc2> &__right)`
- template<class `_CharT2`, class `_Alloc2`>
`rope<_CharT2, _Alloc2> operator+ (const rope<_CharT2, _Alloc2> &__left, const _CharT2 *__right)`
- template<class `_CharT2`, class `_Alloc2`>
`rope<_CharT2, _Alloc2> operator+ (const rope<_CharT2, _Alloc2> &__left, _CharT2 __right)`

5.63.1 Detailed Description

```
template<class _CharT, class _Alloc>
class __gnu_cxx::rope<_CharT, _Alloc>
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation_style.html

Definition at line 327 of file `rope`.

The documentation for this class was generated from the following files:

- `rope`
- `ropeimpl.h`

5.64 `__gnu_cxx::select1st<_Pair>` Struct Template Reference

Inherits `std::_Select1st<_Pair>`.

Public Types

- typedef `_Pair` `argument_type`
- typedef `_Pair::first_type` `result_type`

Public Member Functions

- `_Pair::first_type & operator() (_Pair &__x) const`
- `const _Pair::first_type & operator() (const _Pair &__x) const`
- `template<typename _Pair2 >
_Pair2::first_type & operator() (_Pair2 &__x) const`
- `template<typename _Pair2 >
const _Pair2::first_type & operator() (const _Pair2 &__x) const`

5.64.1 Detailed Description

```
template<class _Pair>
struct __gnu_cxx::select1st< _Pair >
```

An [SGI extension](#) .

Definition at line 200 of file `ext/functional`.

5.64.2 Member Typedef Documentation

5.64.2.1 `typedef _Pair std::unary_function< _Pair , _Pair::first_type >::argument_type` [inherited]

`argument_type` is the type of the argument

Definition at line 108 of file `stl_function.h`.

5.64.2.2 `typedef _Pair::first_type std::unary_function< _Pair , _Pair::first_type >::result_type` [inherited]

`result_type` is the return type

Definition at line 111 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [ext/functional](#)

5.65 `__gnu_cxx::select2nd< _Pair >` Struct Template Reference

Inherits `std::_Select2nd< _Pair >`.

Public Types

- `typedef _Pair argument_type`
- `typedef _Pair::second_type result_type`

Public Member Functions

- `_Pair::second_type & operator() (_Pair &__x) const`
- `const _Pair::second_type & operator() (const _Pair &__x) const`

5.65.1 Detailed Description

```
template<class _Pair>
struct __gnu_cxx::select2nd< _Pair >
```

An [SGL extension](#) .

Definition at line 205 of file `ext/functional`.

5.65.2 Member Typedef Documentation

5.65.2.1 `typedef _Pair std::unary_function< _Pair, _Pair::second_type >::argument_type` `[inherited]`

`argument_type` is the type of the argument

Definition at line 108 of file `stl_function.h`.

5.65.2.2 `typedef _Pair::second_type std::unary_function< _Pair, _Pair::second_type >::result_type` `[inherited]`

`result_type` is the return type

Definition at line 111 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [ext/functional](#)

5.66 `__gnu_cxx::slist<_Tp, _Alloc>` Class Template Reference

Inherits `__gnu_cxx::Slist_base<_Tp, _Alloc>`.

Public Types

- `typedef _Base::allocator_type allocator_type`
- `typedef _Slist_iterator< _Tp, const _Tp &, const _Tp * > const_iterator`
- `typedef const value_type * const_pointer`
- `typedef const value_type & const_reference`
- `typedef ptrdiff_t difference_type`
- `typedef _Slist_iterator< _Tp, _Tp &, _Tp * > iterator`
- `typedef value_type * pointer`
- `typedef value_type & reference`
- `typedef size_t size_type`
- `typedef _Tp value_type`

Public Member Functions

- **slist** (const allocator_type &__a=allocator_type())
- **slist** (size_type __n, const value_type &__x, const allocator_type &__a=allocator_type())
- **slist** (size_type __n)
- template<class _InputIterator >
 slist (_InputIterator __first, _InputIterator __last, const allocator_type &__a=allocator_type())
- **slist** (const **slist** &__x)
- template<class _Integer >
 void **_M_assign_dispatch** (_Integer __n, _Integer __val, __true_type)
- template<class _InputIterator >
 void **_M_assign_dispatch** (_InputIterator __first, _InputIterator __last, __false_type)
- void **_M_fill_assign** (size_type __n, const _Tp &__val)
- void **assign** (size_type __n, const _Tp &__val)
- template<class _InputIterator >
 void **assign** (_InputIterator __first, _InputIterator __last)
- iterator **before_begin** ()
- const_iterator **before_begin** () const
- iterator **begin** ()
- const_iterator **begin** () const
- void **clear** ()
- bool **empty** () const
- iterator **end** ()
- const_iterator **end** () const
- iterator **erase** (iterator __pos)
- iterator **erase** (iterator __first, iterator __last)
- iterator **erase_after** (iterator __pos)
- iterator **erase_after** (iterator __before_first, iterator __last)
- reference **front** ()
- const_reference **front** () const
- allocator_type **get_allocator** () const
- iterator **insert** (iterator __pos, const value_type &__x)
- iterator **insert** (iterator __pos)
- void **insert** (iterator __pos, size_type __n, const value_type &__x)
- template<class _InIterator >
 void **insert** (iterator __pos, _InIterator __first, _InIterator __last)
- iterator **insert_after** (iterator __pos, const value_type &__x)
- iterator **insert_after** (iterator __pos)
- void **insert_after** (iterator __pos, size_type __n, const value_type &__x)
- template<class _InIterator >
 void **insert_after** (iterator __pos, _InIterator __first, _InIterator __last)
- size_type **max_size** () const
- void **merge** (**slist** &__x)
- template<class _StrictWeakOrdering >
 void **merge** (**slist** &, _StrictWeakOrdering)
- **slist** & **operator=** (const **slist** &__x)
- void **pop_front** ()
- iterator **previous** (const_iterator __pos)
- const_iterator **previous** (const_iterator __pos) const
- void **push_front** (const value_type &__x)
- void **push_front** ()

- void **remove** (const _Tp &__val)
- template<class _Predicate >
void **remove_if** (_Predicate __pred)
- void **resize** (size_type new_size, const _Tp &__x)
- void **resize** (size_type new_size)
- void **reverse** ()
- size_type **size** () const
- void **sort** ()
- template<class _StrictWeakOrdering >
void **sort** (_StrictWeakOrdering __comp)
- void **splice** (iterator __pos, [slist](#) &__x)
- void **splice** (iterator __pos, [slist](#) &__x, iterator __i)
- void **splice** (iterator __pos, [slist](#) &__x, iterator __first, iterator __last)
- void **splice_after** (iterator __pos, iterator __before_first, iterator __before_last)
- void **splice_after** (iterator __pos, iterator __prev)
- void **splice_after** (iterator __pos, [slist](#) &__x)
- void **swap** ([slist](#) &__x)
- void **unique** ()
- template<class _BinaryPredicate >
void **unique** (_BinaryPredicate __pred)

Private Types

- typedef _Alloc::template rebind< _Slist_node< _Tp > >::other **_Node_alloc**

Private Member Functions

- _Slist_node_base * **_M_erase_after** (_Slist_node_base * __pos)
- _Slist_node_base * **_M_erase_after** (_Slist_node_base *, _Slist_node_base *)
- _Slist_node< _Tp > * **_M_get_node** ()
- void **_M_put_node** (_Slist_node< _Tp > * __p)

Private Attributes

- _Slist_node_base **_M_head**

5.66.1 Detailed Description

```
template<class _Tp, class _Alloc = allocator<_Tp>>
class __gnu_cxx::slist<_Tp, _Alloc>
```

This is an SGI extension.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation_style.html

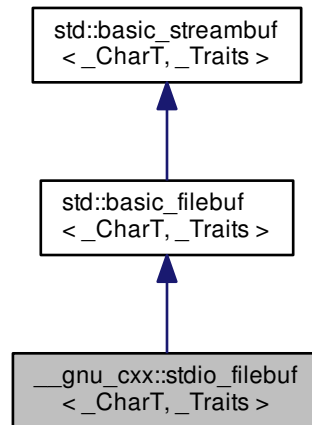
Definition at line 292 of file `slist`.

The documentation for this class was generated from the following file:

- [slist](#)

5.67 `__gnu_cxx::stdio_filebuf<_CharT,_Traits>` Class Template Reference

Inheritance diagram for `__gnu_cxx::stdio_filebuf<_CharT,_Traits>`:



Public Types

- `typedef codecvt< char_type, char, __state_type > __codecvt_type`
- `typedef __basic_file< char > __file_type`
- `typedef basic_filebuf< char_type, traits_type > filebuf_type`
- `typedef traits_type::state_type __state_type`
- `typedef basic_streambuf< char_type, traits_type > __streambuf_type`
- `typedef _CharT char_type`
- `typedef traits_type::int_type int_type`
- `typedef traits_type::off_type off_type`
- `typedef traits_type::pos_type pos_type`
- `typedef std::size_t size_t`
- `typedef _Traits traits_type`

Public Member Functions

- `stdio_filebuf ()`
- `stdio_filebuf (int __fd, std::ios_base::openmode __mode, size_t __size=static_cast< size_t >(BUFSIZ))`
- `stdio_filebuf (std::__c_file * __f, std::ios_base::openmode __mode, size_t __size=static_cast< size_t >(BUFSIZ))`
- `stdio_filebuf (stdio_filebuf &&)=default`
- `virtual ~stdio_filebuf ()`
- `__filebuf_type * close ()`
- `int fd ()`
- `std::__c_file * file ()`

- locale `getloc` () const
- streamsize `in_avail` ()
- bool `is_open` () const throw ()
- `__filebuf_type` * `open` (const char * __s, ios_base::openmode __mode)
- `__filebuf_type` * `open` (const std::string & __s, ios_base::openmode __mode)
- `stdio_filebuf` & `operator=` (`stdio_filebuf` &&)=default
- locale `pubimbue` (const locale & __loc)
- int_type `sbumpc` ()
- int_type `sgetc` ()
- streamsize `sgetn` (char_type * __s, streamsize __n)
- int_type `snextc` ()
- int_type `sputbackc` (char_type __c)
- int_type `sputc` (char_type __c)
- streamsize `sputn` (const char_type * __s, streamsize __n)
- int_type `sungetc` ()
- void `swap` (`stdio_filebuf` & __fb)
- void `swap` (`basic_filebuf` &)
- `basic_streambuf` * `pubsetbuf` (char_type * __s, streamsize __n)
- pos_type `pubseekoff` (off_type __off, ios_base::seekdir __way, ios_base::openmode __mode=ios_base::in|ios_base::out)
- pos_type `pubseekpos` (pos_type __sp, ios_base::openmode __mode=ios_base::in|ios_base::out)
- int `pubsync` ()

Protected Member Functions

- void `__safe_gbump` (streamsize __n)
- void `__safe_pbump` (streamsize __n)
- void `_M_allocate_internal_buffer` ()
- bool `_M_convert_to_external` (char_type *, streamsize)
- void `_M_create_pback` ()
- void `_M_destroy_internal_buffer` () throw ()
- void `_M_destroy_pback` () throw ()
- int `_M_get_ext_pos` (__state_type & __state)
- pos_type `_M_seek` (off_type __off, ios_base::seekdir __way, __state_type __state)
- void `_M_set_buffer` (streamsize __off)
- bool `_M_terminate_output` ()
- void `gbump` (int __n)
- virtual void `imbue` (const locale & __loc)
- virtual int_type `overflow` (int_type __c=_Traits::eof())
- virtual int_type `pbackfail` (int_type __c=_Traits::eof())
- void `pbump` (int __n)
- virtual pos_type `seekoff` (off_type __off, ios_base::seekdir __way, ios_base::openmode __mode=ios_base::in|ios_base::out)
- virtual pos_type `seekpos` (pos_type __pos, ios_base::openmode __mode=ios_base::in|ios_base::out)
- virtual `__streambuf_type` * `setbuf` (char_type * __s, streamsize __n)
- void `setg` (char_type * __gbeg, char_type * __gnext, char_type * __gend)
- void `setp` (char_type * __pbeg, char_type * __pend)
- virtual streamsize `showmanyc` ()

- void **swap** ([basic_streambuf](#) &__sb)
- virtual int [sync](#) ()
- virtual int_type [uflow](#) ()
- virtual int_type [underflow](#) ()
- virtual streamsize [xsgetn](#) (char_type *__s, streamsize __n)
- virtual streamsize [xsputn](#) (const char_type *__s, streamsize __n)

- char_type * [eback](#) () const
- char_type * [gptr](#) () const
- char_type * [egptr](#) () const

- char_type * [pbase](#) () const
- char_type * [pptr](#) () const
- char_type * [epptr](#) () const

Protected Attributes

- char_type * [_M_buf](#)
 - bool [_M_buf_allocated](#)
 - locale [_M_buf_locale](#)
 - size_t [_M_buf_size](#)
 - const [__codecvt_type](#) * [_M_codecvt](#)
 - char * [_M_ext_buf](#)
 - streamsize [_M_ext_buf_size](#)
 - char * [_M_ext_end](#)
 - const char * [_M_ext_next](#)
 - [__file_type](#) [_M_file](#)
 - char_type * [_M_in_beg](#)
 - char_type * [_M_in_cur](#)
 - char_type * [_M_in_end](#)
 - [__c_lock](#) [_M_lock](#)
 - ios_base::openmode [_M_mode](#)
 - char_type * [_M_out_beg](#)
 - char_type * [_M_out_cur](#)
 - char_type * [_M_out_end](#)
 - bool [_M_reading](#)
 - [__state_type](#) [_M_state_beg](#)
 - [__state_type](#) [_M_state_cur](#)
 - [__state_type](#) [_M_state_last](#)
 - bool [_M_writing](#)
-
- char_type [_M_pback](#)
 - char_type * [_M_pback_cur_save](#)
 - char_type * [_M_pback_end_save](#)
 - bool [_M_pback_init](#)

5.67.1 Detailed Description

```
template<typename _CharT, typename _Traits = std::char_traits<_CharT>>
class __gnu_cxx::stdio_filebuf< _CharT, _Traits >
```

Provides a layer of compatibility for C/POSIX.

This GNU extension provides extensions for working with standard C FILE*'s and POSIX file descriptors. It must be instantiated by the user with the type of character used in the file stream, e.g., `stdio_filebuf<char>`.

Definition at line 50 of file `stdio_filebuf.h`.

5.67.2 Constructor & Destructor Documentation

5.67.2.1 `template<typename _CharT, typename _Traits = std::char_traits<_CharT>> __gnu_cxx::stdio_filebuf< _CharT, _Traits>::stdio_filebuf ()` `[inline]`

deferred initialization

Definition at line 65 of file `stdio_filebuf.h`.

References `__gnu_cxx::stdio_filebuf< _CharT, _Traits>::~~stdio_filebuf()`.

Referenced by `__gnu_cxx::stdio_filebuf< _CharT, _Traits>::stdio_filebuf()`, and `__gnu_cxx::stdio_filebuf< _CharT, _Traits>::~~stdio_filebuf()`.

5.67.2.2 `template<typename _CharT, typename _Traits> __gnu_cxx::stdio_filebuf< _CharT, _Traits>::stdio_filebuf (int __fd, std::ios_base::openmode __mode, size_t __size = static_cast<size_t>(BUFSIZ))`

Parameters

<code>__fd</code>	An open file descriptor.
<code>__mode</code>	Same meaning as in a standard filebuf.
<code>__size</code>	Optimal or preferred size of internal buffer, in chars.

This constructor associates a file stream buffer with an open POSIX file descriptor. The file descriptor will be automatically closed when the `stdio_filebuf` is closed/destroyed.

Definition at line 137 of file `stdio_filebuf.h`.

References `std::basic_filebuf< _CharT, _Traits>::_M_buf_size`, `std::basic_filebuf< _CharT, _Traits>::_M_mode`, `std::basic_filebuf< _CharT, _Traits>::_M_reading`, `std::basic_filebuf< _CharT, _Traits>::_M_set_buffer()`, `std::basic_filebuf< _CharT, _Traits>::is_open()`, and `__gnu_cxx::stdio_filebuf< _CharT, _Traits>::stdio_filebuf()`.

5.67.2.3 `template<typename _CharT, typename _Traits> __gnu_cxx::stdio_filebuf< _CharT, _Traits>::stdio_filebuf (std::c_file * __f, std::ios_base::openmode __mode, size_t __size = static_cast<size_t>(BUFSIZ))`

Parameters

<code>__f</code>	An open <code>FILE*</code> .
<code>__mode</code>	Same meaning as in a standard filebuf.
<code>__size</code>	Optimal or preferred size of internal buffer, in chars. Defaults to system's <code>BUFSIZ</code> .

This constructor associates a file stream buffer with an open C `FILE*`. The `FILE*` will not be automatically closed when the `stdio_filebuf` is closed/destroyed.

Definition at line 153 of file `stdio_filebuf.h`.

References `std::basic_filebuf<_CharT, _Traits>::_M_buf_size`, `std::basic_filebuf<_CharT, _Traits>::_M_mode`, `std::basic_filebuf<_CharT, _Traits>::_M_reading`, `std::basic_filebuf<_CharT, _Traits>::_M_set_buffer()`, and `std::basic_filebuf<_CharT, _Traits>::is_open()`.

5.67.2.4 `template<typename _CharT, typename _Traits> __gnu_cxx::stdio_filebuf<_CharT, _Traits>::~~stdio_filebuf ()`
[virtual]

Closes the external data stream if the file descriptor constructor was used.

Definition at line 132 of file `stdio_filebuf.h`.

References `__gnu_cxx::stdio_filebuf<_CharT, _Traits>::stdio_filebuf()`.

Referenced by `__gnu_cxx::stdio_filebuf<_CharT, _Traits>::stdio_filebuf()`.

5.67.3 Member Function Documentation

5.67.3.1 `template<typename _CharT, typename _Traits> void std::basic_filebuf<_CharT, _Traits>::_M_create_pback ()`
[inline], [protected], [inherited]

Initializes pback buffers, and moves normal buffers to safety. Assumptions: `_M_in_cur` has already been moved back

Definition at line 191 of file `fstream`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::pbackfail()`.

5.67.3.2 `template<typename _CharT, typename _Traits> void std::basic_filebuf<_CharT, _Traits>::_M_destroy_pback ()`
`throw)` [inline], [protected], [inherited]

Deactivates pback buffer contents, and restores normal buffer. Assumptions: The pback buffer has only moved forward.

Definition at line 208 of file `fstream`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_filebuf<_CharT, _Traits>::seekoff()`, `std::basic_filebuf<_CharT, _Traits>::seekpos()`, `std::basic_filebuf<_CharT, _Traits>::underflow()`, and `std::basic_filebuf<_CharT, _Traits>::xsgetn()`.

5.67.3.3 `template<typename _CharT, typename _Traits> void std::basic_filebuf<_CharT, _Traits>::_M_set_buffer (streamsize __off) [inline], [protected], [inherited]`

This function sets the pointers of the internal buffer, both get and put areas. Typically:

`__off == egptr() - eback()` upon underflow/uflow (**read** mode); `__off == 0` upon overflow (**write** mode); `__off == -1` upon open, setbuf, seekoff/pos (**uncommitted** mode).

NB: `eptr() - pbase() == _M_buf_size - 1`, since `_M_buf_size` reflects the actual allocated memory and the last cell is reserved for the overflow char of a full put area.

Definition at line 422 of file `fstream`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::close()`, `std::basic_filebuf<_CharT, _Traits>::imbue()`, `std::basic_filebuf<_CharT, _Traits>::open()`, `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_filebuf<_CharT, _Traits>::pbackfail()`, `std::basic_filebuf<_CharT, _Traits>::seekpos()`, `__gnu_cxx::stdio_filebuf<_CharT, _Traits>::stdio_filebuf()`, `std::basic_filebuf<_CharT, _Traits>::underflow()`, `std::basic_filebuf<_CharT, _Traits>::xsgetn()`, and `std::basic_filebuf<_CharT, _Traits>::xsputn()`.

5.67.3.4 `template<typename _CharT, typename _Traits> basic_filebuf<_CharT, _Traits>::_filebuf_type * std::basic_filebuf<_CharT, _Traits>::close () [inherited]`

Closes the currently associated file.

Returns

`this` on success, NULL on failure

If no file is currently open, this function immediately fails.

If a *put buffer area* exists, `overflow(eof)` is called to flush all the characters. The file is then closed.

If any operations fail, this function also fails.

Definition at line 213 of file `fstream.tcc`.

References `std::basic_filebuf<_CharT, _Traits>::_M_mode`, `std::basic_filebuf<_CharT, _Traits>::_M_pback_init`, `std::basic_filebuf<_CharT, _Traits>::_M_reading`, `std::basic_filebuf<_CharT, _Traits>::_M_set_buffer()`, `std::basic_filebuf<_CharT, _Traits>::is_open()`, and `std::basic_filebuf<_CharT, _Traits>::showmanyc()`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::basic_filebuf()`, and `std::basic_filebuf<_CharT, _Traits>::open()`.

5.67.3.5 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::eback () const [inline], [protected], [inherited]`

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Definition at line 482 of file `streambuf`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::imbue()`, `std::basic_filebuf<_CharT, _Traits>::pbackfail()`, `std::basic_filebuf<_CharT, _Traits>::seekpos()`, `std::basic_filebuf<_CharT, _Traits>::underflow()`, and `std::basic_filebuf<_CharT, _Traits>::xsgetn()`.

5.67.3.6 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::egptr ()`
`const [inline], [protected], [inherited]`

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Definition at line 488 of file `streambuf`.

Referenced by `std::ostreambuf_iterator< _CharT, _Traits >::failed()`, `std::basic_filebuf< _CharT, _Traits >::seekpos()`, `std::basic_filebuf< _CharT, _Traits >::showmanyc()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

5.67.3.7 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::eptr ()`
`const [inline], [protected], [inherited]`

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `eptr()` returns the end pointer for the output sequence

Definition at line 535 of file `streambuf`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::xspn()`.

5.67.3.8 `template<typename _CharT, typename _Traits = std::char_traits< _CharT >> int __gnu_cxx::stdio_filebuf< _CharT, _Traits >::fd () [inline]`

Returns

The underlying file descriptor.

Once associated with an external data stream, this function can be used to access the underlying POSIX file descriptor. Note that there is no way for the library to track what you do with the descriptor, so be careful.

Definition at line 118 of file `stdio_filebuf.h`.

5.67.3.9 `template<typename _CharT, typename _Traits = std::char_traits< _CharT >> std::_c_file* __gnu_cxx::stdio_filebuf< _CharT, _Traits >::file () [inline]`

Returns

The underlying FILE*.

This function can be used to access the underlying "C" file pointer. Note that there is no way for the library to track what you do with the file, so be careful.

Definition at line 128 of file `stdio_filebuf.h`.

5.67.3.10 `template<typename _CharT, typename _Traits> void std::basic_streambuf< _CharT, _Traits >::gbump (int __n)`
`[inline], [protected], [inherited]`

Moving the read position.

Parameters

<code>_↔</code>	The delta by which to move.
<code>_n</code>	

This just advances the read position without returning any data.

Definition at line 498 of file `streambuf`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::pbackfail()`, and `std::basic_filebuf<_CharT, _Traits>::xsgetn()`.

5.67.3.11 `template<typename _CharT, typename _Traits> locale std::basic_streambuf<_CharT, _Traits>::getloc () const`
`[inline], [inherited]`

Locale access.

Returns

The current locale in effect.

If `pubimbue(loc)` has been called, then the most recent `loc` is returned. Otherwise the global locale in effect at the time of construction is returned.

Definition at line 226 of file `streambuf`.

5.67.3.12 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::gptr ()`
`const [inline], [protected], [inherited]`

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Definition at line 485 of file `streambuf`.

Referenced by `std::ostreambuf_iterator<_CharT, _Traits>::failed()`, `std::basic_filebuf<_CharT, _Traits>::imbue()`, `std::basic_filebuf<_CharT, _Traits>::pbackfail()`, `std::basic_filebuf<_CharT, _Traits>::seekpos()`, `std::basic_filebuf<_CharT, _Traits>::showmanyc()`, `std::basic_filebuf<_CharT, _Traits>::underflow()`, and `std::basic_filebuf<_CharT, _Traits>::xsgetn()`.

5.67.3.13 `template<typename _CharT, typename _Traits> void std::basic_filebuf<_CharT, _Traits>::imbue (const locale & _loc)`
`[protected], [virtual], [inherited]`

Changes translations.

Parameters

<code>__loc</code>	A new locale.
--------------------	---------------

Translations done during I/O which depend on the current locale are changed by this call. The standard adds, *Between invocations of this function a class derived from `streambuf` can safely cache results of calls to locale functions and to members of facets so obtained.*

Note

Base class version does nothing.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

Definition at line 997 of file `fstream.tcc`.

References `std::basic_filebuf<_CharT, _Traits>::M_ext_buf`, `std::basic_filebuf<_CharT, _Traits>::M_ext_next`, `std::basic_filebuf<_CharT, _Traits>::M_mode`, `std::basic_filebuf<_CharT, _Traits>::M_reading`, `std::basic_filebuf<_CharT, _Traits>::M_set_buffer()`, `std::ios_base::cur`, `std::basic_streambuf<_CharT, _Traits>::eback()`, `std::basic_streambuf<_CharT, _Traits>::gptr()`, `std::basic_filebuf<_CharT, _Traits>::is_open()`, and `std::basic_filebuf<_CharT, _Traits>::seekoff()`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::sync()`.

5.67.3.14 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf<_CharT, _Traits>::in_avail ()`
`[inline], [inherited]`

Looking ahead into the stream.

Returns

The number of characters available.

If a read position is available, returns the number of characters available for reading before the buffer must be refilled. Otherwise returns the derived `showmanyc()`.

Definition at line 284 of file `streambuf`.

5.67.3.15 `template<typename _CharT, typename _Traits> bool std::basic_filebuf<_CharT, _Traits>::is_open () const throw`
`) [inline], [inherited]`

Returns true if the external file is open.

Definition at line 252 of file `fstream`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::close()`, `std::basic_filebuf<_CharT, _Traits>::imbue()`, `std::basic_filebuf<_CharT, _Traits>::open()`, `std::basic_filebuf<_CharT, _Traits>::seekoff()`, `std::basic_filebuf<_CharT, _Traits>::seekpos()`, `std::basic_filebuf<_CharT, _Traits>::setbuf()`, `std::basic_filebuf<_CharT, _Traits>::showmanyc()`, and `__gnu_cxx::stdio_filebuf<_CharT, _Traits>::stdio_filebuf()`.

5.67.3.16 `template<typename _CharT, typename _Traits> basic_filebuf<_CharT, _Traits>::__filebuf_type *`
`std::basic_filebuf<_CharT, _Traits>::open (const char * __s, ios_base::openmode __mode)`
`[inherited]`

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Returns

`this` on success, `NULL` on failure

If a file is already open, this function immediately fails. Otherwise it tries to open the file named `__s` using the flags given in `__mode`.

Table 92, adapted here, gives the relation between openmode combinations and the equivalent `fopen()` flags. (NB: lines `app`, `in|out|app`, `in|app`, `binary|app`, `binary|in|out|app`, and `binary|in|app` per DR 596)

ios_base Flag combination					stdio equivalent
binary	in	out	trunc	app	
		+			w
		+		+	a
				+	a
		+	+		w
	+				r
	+	+			r+
	+	+	+		w+
	+	+		+	a+
	+			+	a+
+		+			wb
+		+		+	ab
+				+	ab
+		+	+		wb
+	+				rb
+	+	+			r+b
+	+	+	+		w+b
+	+	+		+	a+b
+	+			+	a+b

Definition at line 179 of file `fstream.tcc`.

References `std::basic_filebuf<_CharT, _Traits>::_M_mode`, `std::basic_filebuf<_CharT, _Traits>::_M_reading`, `std::basic_filebuf<_CharT, _Traits>::_M_set_buffer()`, `std::ios_base::ate`, `std::basic_filebuf<_CharT, _Traits>::close()`, `std::ios_base::end`, `std::basic_filebuf<_CharT, _Traits>::is_open()`, and `std::basic_filebuf<_CharT, _Traits>::seekoff()`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::basic_filebuf()`.

5.67.3.17 `template<typename _CharT, typename _Traits> __filebuf_type* std::basic_filebuf<_CharT, _Traits>::open (const std::string & __s, ios_base::openmode __mode)` `[inline]`, `[inherited]`

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Returns

`this` on success, `NULL` on failure

Definition at line 307 of file `fstream`.

5.67.3.18 `template<typename _CharT, typename _Traits> basic_filebuf<_CharT, _Traits>::int_type std::basic_filebuf<_CharT, _Traits>::overflow (int_type __c = _Traits::eof()) [protected], [virtual], [inherited]`

Consumes data from the buffer; writes to the controlled sequence.

Parameters

<code>__c</code>	An additional character to consume.
------------------	-------------------------------------

Returns

`eof()` to indicate failure, something else (usually `__c`, or `not_eof()`)

Informally, this function is called when the output buffer is full (or does not exist, as buffering need not actually be done). If a buffer exists, it is *consumed*, with *some effect* on the controlled sequence. (Typically, the buffer is written out to the sequence verbatim.) In either case, the character `c` is also written out, if `__c` is not `eof()`.

For a formal definition of this function, see a good text such as Langer & Kreft, or [27.5.2.4.5]/3-7.

A functioning output streambuf can be created by overriding only this function (no buffer area will be used).

Note

Base class version does nothing, returns `eof()`.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

Definition at line 507 of file `fstream.tcc`.

References `std::basic_filebuf<_CharT, _Traits>::_M_buf_size`, `std::basic_filebuf<_CharT, _Traits>::_M_destroy`, `std::basic_filebuf<_CharT, _Traits>::_M_mode`, `std::basic_filebuf<_CharT, _Traits>::_M_reading`, `std::basic_filebuf<_CharT, _Traits>::_M_set_buffer()`, `std::ios_base::app`, `std::ios_base::cur`, `std::ios_base::out`, `std::basic_streambuf<_CharT, _Traits>::pbase()`, `std::basic_streambuf<_CharT, _Traits>::pbump()`, `std::basic_streambuf<_CharT, _Traits>::pptr()`, `std::basic_filebuf<_CharT, _Traits>::xsgetn()`, and `std::basic_filebuf<_CharT, _Traits>::xspn()`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::pbackfail()`, `std::basic_filebuf<_CharT, _Traits>::seekpos()`, `std::basic_filebuf<_CharT, _Traits>::sync()`, `std::basic_filebuf<_CharT, _Traits>::underflow()`, and `std::basic_filebuf<_CharT, _Traits>::xsgetn()`.

5.67.3.19 `template<typename _CharT, typename _Traits> basic_filebuf<_CharT, _Traits>::int_type std::basic_filebuf<_CharT, _Traits>::pbackfail (int_type __c = _Traits::eof())` `[protected]`, `[virtual]`, `[inherited]`

Tries to back up the input sequence.

Parameters

<code>__c</code>	The character to be inserted back into the sequence.
------------------	--

Returns

`eof()` on failure, *some other value* on success

Postcondition

The constraints of `gptr()`, `eback()`, and `pptr()` are the same as for `underflow()`.

Note

Base class version does nothing, returns `eof()`.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

Definition at line 448 of file `fstream.tcc`.

References `std::basic_filebuf<_CharT, _Traits>::_M_create_pback()`, `std::basic_filebuf<_CharT, _Traits>::_M_←mode`, `std::basic_filebuf<_CharT, _Traits>::_M_pback_init`, `std::basic_filebuf<_CharT, _Traits>::_M_reading`, `std::←basic_filebuf<_CharT, _Traits>::_M_set_buffer()`, `std::ios_base::cur`, `std::basic_streambuf<_CharT, _Traits>←eback()`, `std::basic_streambuf<_CharT, _Traits>::gbump()`, `std::basic_streambuf<_CharT, _Traits>::gptr()`, `std←ios_base::in`, `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_filebuf<_CharT, _Traits>::seekoff()`, and `std::basic_filebuf<_CharT, _Traits>::underflow()`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::underflow()`.

5.67.3.20 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::pbase ()` `const` `[inline]`, `[protected]`, `[inherited]`

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

Definition at line 529 of file `streambuf`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_filebuf<_CharT, _Traits>::seekoff()`, `std←basic_filebuf<_CharT, _Traits>::seekpos()`, `std::basic_filebuf<_CharT, _Traits>::sync()`, and `std::basic_filebuf<_CharT, _Traits>::xspn()`.

5.67.3.21 `template<typename _CharT, typename _Traits> void std::basic_streambuf<_CharT, _Traits>::pbump (int __n)` `[inline]`, `[protected]`, `[inherited]`

Moving the write position.

Parameters

<code>_↔</code>	The delta by which to move.
<code>_n</code>	

This just advances the write position without returning any data.

Definition at line 545 of file streambuf.

Referenced by `std::basic_filebuf<_CharT, _Traits>::overflow()`.

5.67.3.22 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::pptr ()`
`const [inline], [protected], [inherited]`

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

Definition at line 532 of file streambuf.

Referenced by `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_filebuf<_CharT, _Traits>::seekoff()`, `std::basic_filebuf<_CharT, _Traits>::seekpos()`, `std::basic_filebuf<_CharT, _Traits>::sync()`, and `std::basic_filebuf<_CharT, _Traits>::xsputn()`.

5.67.3.23 `template<typename _CharT, typename _Traits> locale std::basic_streambuf<_CharT, _Traits>::pubimbue (const locale &__loc)` `[inline], [inherited]`

Entry point for `imbue()`.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls the derived `imbue(__loc)`.

Definition at line 209 of file streambuf.

5.67.3.24 `template<typename _CharT, typename _Traits> pos_type std::basic_streambuf<_CharT, _Traits>::pubseekoff (off_type __off, ios_base::seekdir __way, ios_base::openmode __mode = ios_base::in | ios_base::out) [inline], [inherited]`

Alters the stream position.

Parameters

<code>__off</code>	Offset.
<code>__way</code>	Value for <code>ios_base::seekdir</code> .
<code>__mode</code>	Value for <code>ios_base::openmode</code> .

Calls virtual seekoff function.

Definition at line 251 of file `streambuf`.

5.67.3.25 `template<typename _CharT, typename _Traits> pos_type std::basic_streambuf<_CharT, _Traits>::pubseekpos (pos_type __sp, ios_base::openmode __mode = ios_base::in | ios_base::out) [inline], [inherited]`

Alters the stream position.

Parameters

<code>__sp</code>	Position
<code>__mode</code>	Value for <code>ios_base::openmode</code> .

Calls virtual seekpos function.

Definition at line 263 of file `streambuf`.

5.67.3.26 `template<typename _CharT, typename _Traits> basic_streambuf* std::basic_streambuf<_CharT, _Traits>::pubsetbuf (char_type *__s, streamsize __n) [inline], [inherited]`

Entry points for derived buffer functions.

The public versions of `pubfoo` dispatch to the protected derived `foo` member functions, passing the arguments (if any) and returning the result unchanged.

Definition at line 239 of file `streambuf`.

5.67.3.27 `template<typename _CharT, typename _Traits> int std::basic_streambuf<_CharT, _Traits>::pubsync () [inline], [inherited]`

Calls virtual sync function.

Definition at line 271 of file `streambuf`.

Referenced by `std::basic_istream<_CharT, _Traits>::sync()`.

5.67.3.28 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf< _CharT, _Traits >::sbumpc ()`
`[inline], [inherited]`

Getting the next character.

Returns

The next character, or eof.

If the input read position is available, returns that character and increments the read pointer, otherwise calls and returns `uflow()`.

Definition at line 316 of file `streambuf`.

Referenced by `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, and `std::istreambuf_iterator< _CharT, _Traits >::operator++()`.

5.67.3.29 `template<typename _CharT, typename _Traits > basic_filebuf< _CharT, _Traits >::pos_type`
`std::basic_filebuf< _CharT, _Traits >::seekoff (off_type, ios_base::seekdir, ios_base::openmode =`
`ios_base::in | ios_base::out) [protected], [virtual], [inherited]`

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented from [std::basic_streambuf< _CharT, _Traits >](#).

Definition at line 800 of file `fstream.tcc`.

References `std::basic_filebuf< _CharT, _Traits >::M_destroy_pback()`, `std::basic_filebuf< _CharT, _Traits >::M_↔_reading`, `std::ios_base::cur`, `std::basic_filebuf< _CharT, _Traits >::is_open()`, `std::basic_streambuf< _CharT, _Traits >::pbase()`, `std::basic_streambuf< _CharT, _Traits >::pptr()`, and `std::basic_filebuf< _CharT, _Traits >::seekpos()`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::imbue()`, `std::basic_filebuf< _CharT, _Traits >::open()`, `std::↔_basic_filebuf< _CharT, _Traits >::pbackfail()`, and `std::basic_filebuf< _CharT, _Traits >::setbuf()`.

5.67.3.30 `template<typename _CharT, typename _Traits > basic_filebuf< _CharT, _Traits >::pos_type std::basic_filebuf<`
`_CharT, _Traits >::seekpos (pos_type, ios_base::openmode = ios_base::in | ios_base::out)`
`[protected], [virtual], [inherited]`

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented from [std::basic_streambuf< _CharT, _Traits >](#).

Definition at line 860 of file `fstream.tcc`.

References `std::basic_filebuf< _CharT, _Traits >::M_destroy_pback()`, `std::basic_filebuf< _CharT, _Traits >::M_↔_ext_buf`, `std::basic_filebuf< _CharT, _Traits >::M_ext_next`, `std::basic_filebuf< _CharT, _Traits >::M_reading`, `std::↔_basic_filebuf< _CharT, _Traits >::M_set_buffer()`, `std::ios_base::beg`, `std::basic_streambuf< _CharT, _Traits >↔_eback()`, `std::basic_streambuf< _CharT, _Traits >::egptr()`, `std::basic_streambuf< _CharT, _Traits >::gptr()`, `std::↔_basic_filebuf< _CharT, _Traits >::is_open()`, `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_streambuf< _CharT, _Traits >::pbase()`, `std::basic_streambuf< _CharT, _Traits >::pptr()`, and `std::basic_filebuf< _CharT, _Traits >::sync()`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::seekoff()`.

5.67.331 `template<typename _CharT, typename _Traits> basic_filebuf<_CharT, _Traits>::__streambuf_type *
std::basic_filebuf<_CharT, _Traits>::setbuf (char_type * __s, streamsize __n)` [protected],
[virtual], [inherited]

Manipulates the buffer.

Parameters

<code>__s</code>	Pointer to a buffer area.
<code>__n</code>	Size of <code>__s</code> .

Returns

`this`

If no file has been opened, and both `__s` and `__n` are zero, then the stream becomes unbuffered. Otherwise, `__s` is used as a buffer; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html#io.streambuf.buffering> for more.

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

Definition at line 771 of file `fstream.tcc`.

References `std::basic_filebuf<_CharT, _Traits>::_M_buf`, `std::basic_filebuf<_CharT, _Traits>::_M_buf_size`, `std::basic_filebuf<_CharT, _Traits>::is_open()`, and `std::basic_filebuf<_CharT, _Traits>::seekoff()`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::xsputn()`.

5.67.332 `template<typename _CharT, typename _Traits> void std::basic_streambuf<_CharT, _Traits>::setg (char_type *
__gbeg, char_type * __gnext, char_type * __gend)` [inline], [protected], [inherited]

Setting the three read area pointers.

Parameters

<code>__gbeg</code>	A pointer.
<code>__gnext</code>	A pointer.
<code>__gend</code>	A pointer.

Postcondition

`__gbeg == eback()`, `__gnext == gptr()`, and `__gend == egptr()`

Definition at line 509 of file `streambuf`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::xsgetn()`.

5.67.3.33 `template<typename _CharT, typename _Traits> void std::basic_streambuf<_CharT, _Traits>::setp (char_type * __pbeg, char_type * __pend) [inline], [protected], [inherited]`

Setting the three write area pointers.

Parameters

<code>__pbeg</code>	A pointer.
<code>__pend</code>	A pointer.

Postcondition

`__pbeg == pbase()`, `__pbeg == pptr()`, and `__pend == epptr()`

Definition at line 555 of file streambuf.

5.67.3.34 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::sgetc () [inline], [inherited]`

Getting the next character.

Returns

The next character, or eof.

If the input read position is available, returns that character, otherwise calls and returns `underflow()`. Does not move the read position after fetching the character.

Definition at line 338 of file streambuf.

Referenced by `std::istreambuf_iterator<_CharT, _Traits>::equal()`, `std::ostreambuf_iterator<_CharT, _Traits>::failed()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, and `std::basic_streambuf<_CharT, _Traits>::xsputn()`.

5.67.3.35 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf<_CharT, _Traits>::sgetn (char_type * __s, streamsize __n) [inline], [inherited]`

Entry point for `xsgetn`.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	A count.

Returns `xsgntrn(__s, __n)`. The effect is to fill `__s[0]` through `__s[__n-1]` with characters from the input sequence, if possible.

Definition at line 357 of file `streambuf`.

5.67.3.36 `template<typename _CharT, typename _Traits> streamsize std::basic_filebuf<_CharT, _Traits>::showmanyc ()`
`[protected], [virtual], [inherited]`

Investigating the data available.

Returns

An estimate of the number of characters available in the input sequence, or -1.

If it returns a positive value, then successive calls to `underflow()` will not return `traits::eof()` until at least that number of characters have been supplied. If `showmanyc()` returns -1, then calls to `underflow()` or `uflow()` will fail. [27.5.2.4.3]/1

Note

Base class version does nothing, returns zero.

The standard adds that *the intention is not only that the calls [to `underflow` or `uflow`] will not return `eof()` but that they will return immediately.*

The standard adds that *the morphemes of `showmanyc` are **es-how-many-see**, not **show-manic**.*

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

Definition at line 263 of file `fstream.tcc`.

References `std::basic_filebuf<_CharT, _Traits>::M_mode`, `std::ios_base::binary`, `std::basic_streambuf<_CharT, _Traits>::egptr()`, `std::basic_streambuf<_CharT, _Traits>::gptr()`, `std::ios_base::in`, `std::basic_filebuf<_CharT, _Traits>::is_open()`, and `std::basic_filebuf<_CharT, _Traits>::underflow()`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::close()`.

5.67.3.37 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::snextc ()`
`[inline], [inherited]`

Getting the next character.

Returns

The next character, or `eof`.

Calls `sbumpc()`, and if that function returns `traits::eof()`, so does this function. Otherwise, `sgetc()`.

Definition at line 298 of file `streambuf`.

Referenced by `std::ostreambuf_iterator<_CharT, _Traits>::failed()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, and `std::basic_streambuf<_CharT, _Traits>::xspntrn()`.

5.67.3.38 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::sputbackc (`
`char_type __c) [inline], [inherited]`

Pushing characters back into the input stream.

Parameters

<code>__c</code>	The character to push back.
------------------	-----------------------------

Returns

The previous character, if possible.

Similar to `sungetc()`, but `__c` is pushed onto the stream instead of *the previous character*. If successful, the next character fetched from the input stream will be `__c`.

Definition at line 372 of file `streambuf`.

Referenced by `std::basic_istream<_CharT, _Traits>::putback()`.

5.67.3.39 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::sputc (char_type __c) [inline], [inherited]`

Entry point for all single-character output functions.

Parameters

<code>__c</code>	A character to output.
------------------	------------------------

Returns

`__c`, if possible.

One of two public output functions.

If a write position is available for the output sequence (i.e., the buffer is not full), stores `__c` in that position, increments the position, and returns `traits::to_int_type(__c)`. If a write position is not available, returns `overflow(__c)`.

Definition at line 424 of file `streambuf`.

Referenced by `std::basic_istream<_CharT, _Traits>::get()`, `std::ostreambuf_iterator<_CharT, _Traits>::operator=()`, and `std::basic_streambuf<_CharT, _Traits>::xsputn()`.

5.67.3.40 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf<_CharT, _Traits>::sputn (const char_type * __s, streamsize __n) [inline], [inherited]`

Entry point for all single-character output functions.

Parameters

<code>__s</code>	A buffer read area.
<code>__n</code>	A count.

One of two public output functions.

Returns `xspn(__s, __n)`. The effect is to write `__s[0]` through `__s[__n-1]` to the output sequence, if possible.

Definition at line 450 of file `streambuf`.

Referenced by `std::ostreambuf_iterator<_CharT, _Traits>::failed()`.

5.67.3.41 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::sungetc ()`
`[inline], [inherited]`

Moving backwards in the input stream.

Returns

The previous character, if possible.

If a putback position is available, this function decrements the input pointer and returns that character. Otherwise, calls and returns `pbackfail()`. The effect is to *unget* the last character *gotten*.

Definition at line 397 of file `streambuf`.

Referenced by `std::basic_istream<_CharT, _Traits>::unget()`.

5.67.3.42 `template<typename _CharT, typename _Traits> int std::basic_filebuf<_CharT, _Traits>::sync ()`
`[protected], [virtual], [inherited]`

Synchronizes the buffer arrays with the controlled sequences.

Returns

-1 on failure.

Each derived class provides its own appropriate behavior, including the definition of *failure*.

Note

Base class version does nothing, returns zero.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

Definition at line 980 of file `fstream.tcc`.

References `std::basic_filebuf<_CharT, _Traits>::imbue()`, `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_streambuf<_CharT, _Traits>::pbase()`, and `std::basic_streambuf<_CharT, _Traits>::pptr()`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::seekpos()`.

5.67.3.43 `template<typename _CharT, typename _Traits> virtual int_type std::basic_streambuf< _CharT, _Traits >::uflow ()`
`[inline], [protected], [virtual], [inherited]`

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function does the same thing as `underflow()`, and in fact is required to call that function. It also returns the new character, like `underflow()` does. However, this function also moves the read position forward by one.

Reimplemented in [__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >](#).

Definition at line 700 of file `streambuf`.

5.67.3.44 `template<typename _CharT, typename _Traits> basic_filebuf< _CharT, _Traits >::int_type std::basic_filebuf< _CharT, _Traits >::underflow ()`
`[protected], [virtual], [inherited]`

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function is called when the input buffer is exhausted (or does not exist, as buffering need not actually be done). If a buffer exists, it is *refilled*. In either case, the next available character is returned, or `traits::eof()` to indicate a null pending sequence.

For a formal definition of the pending sequence, see a good text such as Langer & Kreft, or [27.5.2.4.3]/7-14.

A functioning input streambuf can be created by overriding only this function (no buffer area will be used). For an example, see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html>

Note

Base class version does nothing, returns `eof()`.

Reimplemented from [std::basic_streambuf< _CharT, _Traits >](#).

Definition at line 289 of file `fstream.tcc`.

References `std::basic_filebuf< _CharT, _Traits >::_M_buf_size`, `std::basic_filebuf< _CharT, _Traits >::_M_destroy`, `↵`
`pback()`, `std::basic_filebuf< _CharT, _Traits >::_M_ext_buf`, `std::basic_filebuf< _CharT, _Traits >::_M_ext_buf_size`,
`std::basic_filebuf< _CharT, _Traits >::_M_ext_next`, `std::basic_filebuf< _CharT, _Traits >::_M_mode`, `std::basic_↵`
`filebuf< _CharT, _Traits >::_M_reading`, `std::basic_filebuf< _CharT, _Traits >::_M_set_buffer()`, `std::basic_streambuf<`
`_CharT, _Traits >::eback()`, `std::basic_streambuf< _CharT, _Traits >::egptr()`, `std::basic_streambuf< _CharT, _Traits`
`>::gptr()`, `std::ios_base::in`, `std::min()`, `std::basic_filebuf< _CharT, _Traits >::overflow()`, and `std::basic_filebuf< _CharT,`
`_Traits >::pbackfail()`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, and `std::basic_filebuf< _CharT, _Traits >↵`
`::showmanyc()`.

5.67.3.45 `template<typename _CharT, typename _Traits> streamsize std::basic_filebuf< _CharT, _Traits >::xsgetn (`
`char_type * __s, streamsize __n)` `[protected], [virtual], [inherited]`

Multiple character extraction.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to assign.

Returns

The number of characters assigned.

Fills `__s[0]` through `__s[__n-1]` with characters from the input sequence, as if by `sbumpc()`. Stops when either `__n` characters have been copied, or when `traits::eof()` would be copied.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

Definition at line 635 of file `fstream.tcc`.

References `std::basic_filebuf<_CharT, _Traits>::_M_buf_size`, `std::basic_filebuf<_CharT, _Traits>::_M_destroy`, `std::basic_filebuf<_CharT, _Traits>::_M_mode`, `std::basic_filebuf<_CharT, _Traits>::_M_pback_init`, `std::basic_filebuf<_CharT, _Traits>::_M_reading`, `std::basic_filebuf<_CharT, _Traits>::_M_set_buffer()`, `std::basic_streambuf<_CharT, _Traits>::eback()`, `std::basic_streambuf<_CharT, _Traits>::egptr()`, `std::basic_streambuf<_CharT, _Traits>::gbump()`, `std::basic_streambuf<_CharT, _Traits>::gptr()`, `std::ios_base::in`, `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_streambuf<_CharT, _Traits>::setg()`, `std::basic_streambuf<_CharT, _Traits>::xsgetn()`, and `std::basic_filebuf<_CharT, _Traits>::xsputn()`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::overflow()`.

5.67.3.46 `template<typename _CharT, typename _Traits> streamsize std::basic_filebuf<_CharT, _Traits>::xsputn (const char_type * __s, streamsize __n)` `[protected]`, `[virtual]`, `[inherited]`

Multiple character insertion.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to write.

Returns

The number of characters written.

Writes `__s[0]` through `__s[__n-1]` to the output sequence, as if by `sputc()`. Stops when either `n` characters have been copied, or when `sputc()` would return `traits::eof()`.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented from `std::basic_streambuf< _CharT, _Traits >`.

Definition at line 723 of file `fstream.tcc`.

References `std::basic_filebuf< _CharT, _Traits >::_M_buf_size`, `std::basic_filebuf< _CharT, _Traits >::_M_mode`, `std::basic_filebuf< _CharT, _Traits >::_M_reading`, `std::basic_filebuf< _CharT, _Traits >::_M_set_buffer()`, `std::ios_base::app`, `std::basic_streambuf< _CharT, _Traits >::epptr()`, `std::min()`, `std::ios_base::out`, `std::basic_streambuf< _CharT, _Traits >::pbase()`, `std::basic_streambuf< _CharT, _Traits >::pptr()`, `std::basic_filebuf< _CharT, _Traits >::setbuf()`, and `std::basic_streambuf< _CharT, _Traits >::xsputn()`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

5.67.4 Member Data Documentation

5.67.4.1 `template<typename _CharT, typename _Traits> char_type* std::basic_filebuf< _CharT, _Traits >::_M_buf`
[protected], [inherited]

Pointer to the beginning of internal buffer.

Definition at line 128 of file `fstream`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::basic_filebuf()`, and `std::basic_filebuf< _CharT, _Traits >::setbuf()`.

5.67.4.2 `template<typename _CharT, typename _Traits> locale std::basic_streambuf< _CharT, _Traits >::_M_buf_locale`
[protected], [inherited]

Current locale setting.

Definition at line 192 of file `streambuf`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::basic_filebuf()`.

5.67.4.3 `template<typename _CharT, typename _Traits> size_t std::basic_filebuf< _CharT, _Traits >::_M_buf_size`
[protected], [inherited]

Actual size of internal buffer. This number is equal to the size of the put area + 1 position, reserved for the overflow char of a full area.

Definition at line 135 of file `fstream`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::basic_filebuf()`, `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_filebuf< _CharT, _Traits >::setbuf()`, `__gnu_cxx::stdio_filebuf< _CharT, _Traits >::stdio_filebuf()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, `std::basic_filebuf< _CharT, _Traits >::xsgetn()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

5.67.4.4 `template<typename _CharT, typename _Traits> char* std::basic_filebuf<_CharT, _Traits>::_M_ext_buf`
`[protected], [inherited]`

Buffer for external characters. Used for input when `codecvt::always_noconv() == false`. When valid, this corresponds to `eback()`.

Definition at line 170 of file `fstream`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::basic_filebuf()`, `std::basic_filebuf<_CharT, _Traits>::imbue()`, `std::basic_filebuf<_CharT, _Traits>::seekpos()`, and `std::basic_filebuf<_CharT, _Traits>::underflow()`.

5.67.4.5 `template<typename _CharT, typename _Traits> streamsize std::basic_filebuf<_CharT, _Traits>::_M_ext_buf_size`
`[protected], [inherited]`

Size of buffer held by `_M_ext_buf`.

Definition at line 175 of file `fstream`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::basic_filebuf()`, and `std::basic_filebuf<_CharT, _Traits>::underflow()`.

5.67.4.6 `template<typename _CharT, typename _Traits> const char* std::basic_filebuf<_CharT, _Traits>::_M_ext_next`
`[protected], [inherited]`

Pointers into the buffer held by `_M_ext_buf` that delimit a subsequence of bytes that have been read but not yet converted. When valid, `_M_ext_next` corresponds to `egptr()`.

Definition at line 182 of file `fstream`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::basic_filebuf()`, `std::basic_filebuf<_CharT, _Traits>::imbue()`, `std::basic_filebuf<_CharT, _Traits>::seekpos()`, and `std::basic_filebuf<_CharT, _Traits>::underflow()`.

5.67.4.7 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::_M_in_beg`
`[protected], [inherited]`

Start of get area.

Definition at line 184 of file `streambuf`.

5.67.4.8 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::_M_in_cur`
`[protected], [inherited]`

Current read area.

Definition at line 185 of file `streambuf`.

5.67.4.9 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::_M_in_end`
`[protected], [inherited]`

End of get area.

Definition at line 186 of file `streambuf`.

5.67.4.10 `template<typename _CharT, typename _Traits> ios_base::openmode std::basic_filebuf< _CharT, _Traits >::_M_mode` `[protected], [inherited]`

Place to stash in || out || in | out settings for current filebuf.

Definition at line 113 of file fstream.

Referenced by `std::basic_filebuf< _CharT, _Traits >::basic_filebuf()`, `std::basic_filebuf< _CharT, _Traits >::close()`, `std::basic_filebuf< _CharT, _Traits >::imbue()`, `std::basic_filebuf< _CharT, _Traits >::open()`, `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, `std::basic_filebuf< _CharT, _Traits >::showmanyc()`, `__gnu_cxx::stdio_filebuf< _CharT, _Traits >::stdio_filebuf()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, `std::basic_filebuf< _CharT, _Traits >::xsgetn()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

5.67.4.11 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_beg` `[protected], [inherited]`

Start of put area.

Definition at line 187 of file streambuf.

5.67.4.12 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_cur` `[protected], [inherited]`

Current put area.

Definition at line 188 of file streambuf.

5.67.4.13 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_end` `[protected], [inherited]`

End of put area.

Definition at line 189 of file streambuf.

5.67.4.14 `template<typename _CharT, typename _Traits> char_type std::basic_filebuf< _CharT, _Traits >::_M_pback` `[protected], [inherited]`

Necessary bits for putback buffer management.

Note

pbacks of over one character are not currently supported.

Definition at line 156 of file fstream.

Referenced by `std::basic_filebuf< _CharT, _Traits >::basic_filebuf()`.

5.67.4.15 `template<typename _CharT, typename _Traits> char_type* std::basic_filebuf<_CharT, _Traits>::_M_pback_cur_save` [protected], [inherited]

Necessary bits for putback buffer management.

Note

pbacks of over one character are not currently supported.

Definition at line 157 of file `fstream`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::basic_filebuf()`.

5.67.4.16 `template<typename _CharT, typename _Traits> char_type* std::basic_filebuf<_CharT, _Traits>::_M_pback_end_save` [protected], [inherited]

Necessary bits for putback buffer management.

Note

pbacks of over one character are not currently supported.

Definition at line 158 of file `fstream`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::basic_filebuf()`.

5.67.4.17 `template<typename _CharT, typename _Traits> bool std::basic_filebuf<_CharT, _Traits>::_M_pback_init` [protected], [inherited]

Necessary bits for putback buffer management.

Note

pbacks of over one character are not currently supported.

Definition at line 159 of file `fstream`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::basic_filebuf()`, `std::basic_filebuf<_CharT, _Traits>::close()`, `std::basic_filebuf<_CharT, _Traits>::pbackfail()`, and `std::basic_filebuf<_CharT, _Traits>::xsgetn()`.

5.67.4.18 `template<typename _CharT, typename _Traits> bool std::basic_filebuf<_CharT, _Traits>::_M_reading` [protected], [inherited]

`_M_reading == false` && `_M_writing == false` for **uncommitted** mode; `_M_reading == true` for **read** mode; `_M_writing == true` for **write** mode;

NB: `_M_reading == true` && `_M_writing == true` is unused.

Definition at line 147 of file `fstream`.

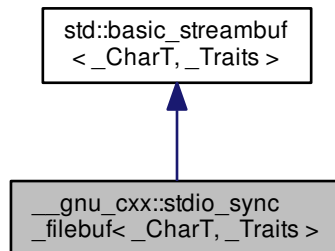
Referenced by `std::basic_filebuf<_CharT, _Traits>::basic_filebuf()`, `std::basic_filebuf<_CharT, _Traits>::close()`, `std::basic_filebuf<_CharT, _Traits>::imbue()`, `std::basic_filebuf<_CharT, _Traits>::open()`, `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_filebuf<_CharT, _Traits>::pbackfail()`, `std::basic_filebuf<_CharT, _Traits>::seekoff()`, `std::basic_filebuf<_CharT, _Traits>::seekpos()`, `__gnu_cxx::stdio_filebuf<_CharT, _Traits>::stdio_filebuf()`, `std::basic_filebuf<_CharT, _Traits>::underflow()`, `std::basic_filebuf<_CharT, _Traits>::xsgetn()`, and `std::basic_filebuf<_CharT, _Traits>::xspn()`.

The documentation for this class was generated from the following file:

- [stdio_filebuf.h](#)

5.68 `__gnu_cxx::stdio_sync_filebuf<_CharT,_Traits>` Class Template Reference

Inheritance diagram for `__gnu_cxx::stdio_sync_filebuf<_CharT,_Traits>`:



Public Types

- typedef `_CharT` **char_type**
- typedef `traits_type::int_type` **int_type**
- typedef `traits_type::off_type` **off_type**
- typedef `traits_type::pos_type` **pos_type**
- typedef `_Traits` **traits_type**

Public Member Functions

- **stdio_sync_filebuf** (`std::__c_file * __f`)
- **stdio_sync_filebuf** (`stdio_sync_filebuf && __fb`) noexcept
- `std::__c_file * file` ()
- locale `getloc` () const
- streamsize `in_avail` ()
- **stdio_sync_filebuf & operator=** (`stdio_sync_filebuf && __fb`) noexcept
- locale `pubimbue` (const locale & __loc)
- `int_type sbumpc` ()
- `int_type sgetc` ()
- streamsize `sgetn` (`char_type * __s`, streamsize __n)
- `int_type snextc` ()
- `int_type sputbackc` (`char_type __c`)
- `int_type sputc` (`char_type __c`)
- streamsize `sputn` (const `char_type * __s`, streamsize __n)
- `int_type sungetc` ()
- void **swap** (`stdio_sync_filebuf & __fb`)
- `basic_streambuf * pubsetbuf` (`char_type * __s`, streamsize __n)
- `pos_type pubseekoff` (`off_type __off`, `ios_base::seekdir __way`, `ios_base::openmode __mode=ios_base::in|ios_base::out`)
- `pos_type pubseekpos` (`pos_type __sp`, `ios_base::openmode __mode=ios_base::in|ios_base::out`)
- `int pubsync` ()

Protected Member Functions

- void `__safe_gbump` (streamsize __n)
- void `__safe_pbump` (streamsize __n)
- void `gbump` (int __n)
- virtual void `imbue` (const locale &__loc)
- virtual `int_type` `overflow` (`int_type` __c=traits_type::eof())
- virtual `int_type` `pbackfail` (`int_type` __c=traits_type::eof())
- void `pbump` (int __n)
- virtual `std::streampos` `seekoff` (`std::streamoff` __off, `std::ios_base::seekdir` __dir, `std::ios_base::openmode`=`std::ios_base::in`|`std::ios_base::out`)
- virtual `pos_type` `seekoff` (`off_type`, `ios_base::seekdir`, `ios_base::openmode`=`ios_base::in`|`ios_base::out`)
- virtual `std::streampos` `seekpos` (`std::streampos` __pos, `std::ios_base::openmode` __mode=`std::ios_base::in`|`std::ios_base::out`)
- virtual `pos_type` `seekpos` (`pos_type`, `ios_base::openmode`=`ios_base::in`|`ios_base::out`)
- virtual `basic_streambuf<char_type, _Traits> *` `setbuf` (`char_type *`__, streamsize)
- void `setg` (`char_type *`__gbeg, `char_type *`__gnext, `char_type *`__gend)
- void `setp` (`char_type *`__pbeg, `char_type *`__pend)
- virtual streamsize `showmanyc` ()
- void `swap` (`basic_streambuf` &__sb)
- virtual int `sync` ()
- `int_type` `syncgetc` ()
- template<>
 `stdio_sync_filebuf<char>::int_type` `syncgetc` ()
- template<>
 `stdio_sync_filebuf<wchar_t>::int_type` `syncgetc` ()
- `int_type` `syncputc` (`int_type` __c)
- template<>
 `stdio_sync_filebuf<char>::int_type` `syncputc` (`int_type` __c)
- template<>
 `stdio_sync_filebuf<wchar_t>::int_type` `syncputc` (`int_type` __c)
- `int_type` `syncungetc` (`int_type` __c)
- template<>
 `stdio_sync_filebuf<char>::int_type` `syncungetc` (`int_type` __c)
- template<>
 `stdio_sync_filebuf<wchar_t>::int_type` `syncungetc` (`int_type` __c)
- virtual `int_type` `uflow` ()
- virtual `int_type` `underflow` ()
- virtual `std::streamsize` `xsggetn` (`char_type *`__s, `std::streamsize` __n)
- template<>
 `std::streamsize` `xsggetn` (`char *`__s, `std::streamsize` __n)
- template<>
 `std::streamsize` `xsggetn` (`wchar_t *`__s, `std::streamsize` __n)
- virtual streamsize `xsggetn` (`char_type *`__s, streamsize __n)
- virtual `std::streamsize` `xsputn` (const `char_type *`__s, `std::streamsize` __n)
- template<>
 `std::streamsize` `xsputn` (const `char *`__s, `std::streamsize` __n)
- template<>
 `std::streamsize` `xsputn` (const `wchar_t *`__s, `std::streamsize` __n)
- virtual streamsize `xsputn` (const `char_type *`__s, streamsize __n)

- `char_type * eback () const`
- `char_type * gptr () const`
- `char_type * egptr () const`
- `char_type * pbase () const`
- `char_type * pptr () const`
- `char_type * epptr () const`

Protected Attributes

- `locale _M_buf_locale`
- `char_type * _M_in_beg`
- `char_type * _M_in_cur`
- `char_type * _M_in_end`
- `char_type * _M_out_beg`
- `char_type * _M_out_cur`
- `char_type * _M_out_end`

5.68.1 Detailed Description

```
template<typename _CharT, typename _Traits = std::char_traits<_CharT>>
class __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >
```

Provides a layer of compatibility for C.

This GNU extension provides extensions for working with standard C FILE*'s. It must be instantiated by the user with the type of character used in the file stream, e.g., `stdio_filebuf<char>`.

Definition at line 57 of file `stdio_sync_filebuf.h`.

5.68.2 Member Function Documentation

5.68.2.1 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::eback ()`
`const [inline], [protected], [inherited]`

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Definition at line 482 of file `streambuf`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::imbue()`, `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, `std::basic_filebuf< _CharT, _Traits >::seekpos()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

5.68.2.2 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::egptr ()`
`const [inline], [protected], [inherited]`

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Definition at line 488 of file `streambuf`.

Referenced by `std::ostreambuf_iterator<_CharT, _Traits>::failed()`, `std::basic_filebuf<_CharT, _Traits>::seekpos()`, `std::basic_filebuf<_CharT, _Traits>::showmanyc()`, `std::basic_filebuf<_CharT, _Traits>::underflow()`, and `std::basic_filebuf<_CharT, _Traits>::xsgetn()`.

5.68.2.3 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::epptr ()`
`const [inline], [protected], [inherited]`

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

Definition at line 535 of file `streambuf`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::xspn()`.

5.68.2.4 `template<typename _CharT, typename _Traits = std::char_traits<_CharT>> std::__c_file*`
`__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>::file () [inline]`

Returns

The underlying `FILE*`.

This function can be used to access the underlying C file pointer. Note that there is no way for the library to track what you do with the file, so be careful.

Definition at line 118 of file `stdio_sync_filebuf.h`.

5.68.2.5 `template<typename _CharT, typename _Traits> void std::basic_streambuf<_CharT, _Traits>::gbump (int __n)`
`[inline], [protected], [inherited]`

Moving the read position.

Parameters

<code>_↔</code>	The delta by which to move.
<code>_n</code>	

This just advances the read position without returning any data.

Definition at line 498 of file streambuf.

Referenced by `std::basic_filebuf<_CharT, _Traits>::pbackfail()`, and `std::basic_filebuf<_CharT, _Traits>::xsgetn()`.

5.68.2.6 `template<typename _CharT, typename _Traits> locale std::basic_streambuf<_CharT, _Traits>::getloc () const`
`[inline], [inherited]`

Locale access.

Returns

The current locale in effect.

If `pubimbue(loc)` has been called, then the most recent `loc` is returned. Otherwise the global locale in effect at the time of construction is returned.

Definition at line 226 of file streambuf.

5.68.2.7 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::gptr ()`
`const [inline], [protected], [inherited]`

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Definition at line 485 of file streambuf.

Referenced by `std::ostreambuf_iterator<_CharT, _Traits>::failed()`, `std::basic_filebuf<_CharT, _Traits>::imbue()`, `std::basic_filebuf<_CharT, _Traits>::pbackfail()`, `std::basic_filebuf<_CharT, _Traits>::seekpos()`, `std::basic_filebuf<_CharT, _Traits>::showmanyc()`, `std::basic_filebuf<_CharT, _Traits>::underflow()`, and `std::basic_filebuf<_CharT, _Traits>::xsgetn()`.

5.68.2.8 `template<typename _CharT, typename _Traits> virtual void std::basic_streambuf<_CharT, _Traits>::imbue (const`
`locale &__loc) [inline], [protected], [virtual], [inherited]`

Changes translations.

Parameters

<code>__loc</code>	A new locale.
--------------------	---------------

Translations done during I/O which depend on the current locale are changed by this call. The standard adds, *Between invocations of this function a class derived from `streambuf` can safely cache results of calls to locale functions and to members of facets so obtained.*

Note

Base class version does nothing.

Reimplemented in [std::basic_filebuf<_CharT, _Traits>](#), [std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>](#), and [std::basic_filebuf<char_type, traits_type>](#).

Definition at line 576 of file `streambuf`.

5.68.2.9 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf<_CharT, _Traits>::in_avail ()`
`[inline], [inherited]`

Looking ahead into the stream.

Returns

The number of characters available.

If a read position is available, returns the number of characters available for reading before the buffer must be refilled. Otherwise returns the derived `showmanyc()`.

Definition at line 284 of file `streambuf`.

5.68.2.10 `template<typename _CharT, typename _Traits = std::char_traits<_CharT>> virtual int_type`
`__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>::overflow (int_type __c = traits_type::eof())`
`[inline], [protected], [virtual]`

Consumes data from the buffer; writes to the controlled sequence.

Parameters

<code>__c</code>	An additional character to consume.
------------------	-------------------------------------

Returns

`eof()` to indicate failure, something else (usually `__c`, or `not_eof()`)

Informally, this function is called when the output buffer is full (or does not exist, as buffering need not actually be done). If a buffer exists, it is *consumed*, with *some effect* on the controlled sequence. (Typically, the buffer is written out to the sequence verbatim.) In either case, the character `c` is also written out, if `__c` is not `eof()`.

For a formal definition of this function, see a good text such as Langer & Kreft, or [27.5.2.4.5]/3-7.

A functioning output streambuf can be created by overriding only this function (no buffer area will be used).

Note

Base class version does nothing, returns eof().

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

Definition at line 171 of file `stdio_sync_filebuf.h`.

```
5.68.2.11 template<typename _CharT, typename _Traits = std::char_traits<_CharT>> virtual int_type
    _gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>::pbackfail( int_type __c=traits_type::eof() )
    [inline], [protected], [virtual]
```

Tries to back up the input sequence.

Parameters

<code>__c</code>	The character to be inserted back into the sequence.
------------------	--

Returns

eof() on failure, *some other value* on success

Postcondition

The constraints of `gptr()`, `eback()`, and `pptr()` are the same as for `underflow()`.

Note

Base class version does nothing, returns eof().

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

Definition at line 146 of file `stdio_sync_filebuf.h`.

```
5.68.2.12 template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::pbase ( )
    const [inline], [protected], [inherited]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

Definition at line 529 of file `streambuf`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_filebuf<_CharT, _Traits>::seekoff()`, `std::basic_filebuf<_CharT, _Traits>::seekpos()`, `std::basic_filebuf<_CharT, _Traits>::sync()`, and `std::basic_filebuf<_CharT, _Traits>::xsputn()`.

5.68.2.13 `template<typename _CharT, typename _Traits> void std::basic_streambuf<_CharT, _Traits>::pbump (int __n)`
`[inline], [protected], [inherited]`

Moving the write position.

Parameters

<code>__↔ __n</code>	The delta by which to move.
--------------------------	-----------------------------

This just advances the write position without returning any data.

Definition at line 545 of file `streambuf`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::overflow()`.

5.68.2.14 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::pptr ()`
`const [inline], [protected], [inherited]`

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

Definition at line 532 of file `streambuf`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_filebuf<_CharT, _Traits>::seekoff()`, `std::basic_filebuf<_CharT, _Traits>::seekpos()`, `std::basic_filebuf<_CharT, _Traits>::sync()`, and `std::basic_filebuf<_CharT, _Traits>::xspn()`.

5.68.2.15 `template<typename _CharT, typename _Traits> locale std::basic_streambuf<_CharT, _Traits>::pubimbue (const locale & __loc)`
`[inline], [inherited]`

Entry point for `imbue()`.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls the derived `imbue(__loc)`.

Definition at line 209 of file `streambuf`.

```
5.68.2.16  template<typename _CharT, typename _Traits> pos_type std::basic_streambuf<_CharT, _Traits>::pubseekoff (
    off_type __off, ios_base::seekdir __way, ios_base::openmode __mode = ios_base::in | ios_base::out )
    [inline], [inherited]
```

Alters the stream position.

Parameters

<code>__off</code>	Offset.
<code>__way</code>	Value for <code>ios_base::seekdir</code> .
<code>__mode</code>	Value for <code>ios_base::openmode</code> .

Calls virtual `seekoff` function.

Definition at line 251 of file `streambuf`.

```
5.68.2.17  template<typename _CharT, typename _Traits> pos_type std::basic_streambuf<_CharT, _Traits>::pubseekpos
    ( pos_type __sp, ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline],
    [inherited]
```

Alters the stream position.

Parameters

<code>__sp</code>	Position
<code>__mode</code>	Value for <code>ios_base::openmode</code> .

Calls virtual `seekpos` function.

Definition at line 263 of file `streambuf`.

```
5.68.2.18  template<typename _CharT, typename _Traits> basic_streambuf* std::basic_streambuf<_CharT, _Traits>
    >::pubsetbuf( char_type* __s, streamsize __n ) [inline], [inherited]
```

Entry points for derived buffer functions.

The public versions of `pubfoo` dispatch to the protected derived `foo` member functions, passing the arguments (if any) and returning the result unchanged.

Definition at line 239 of file `streambuf`.

```
5.68.2.19  template<typename _CharT, typename _Traits> int std::basic_streambuf<_CharT, _Traits>::pubsync ( )
    [inline], [inherited]
```

Calls virtual `sync` function.

Definition at line 271 of file `streambuf`.

Referenced by `std::basic_istream<_CharT, _Traits>::sync()`.

5.68.2.20 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::sbumpc ()`
`[inline], [inherited]`

Getting the next character.

Returns

The next character, or eof.

If the input read position is available, returns that character and increments the read pointer, otherwise calls and returns `uflow()`.

Definition at line 316 of file `streambuf`.

Referenced by `std::basic_istream<_CharT, _Traits>::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, and `std::istreambuf_iterator<_CharT, _Traits>::operator++()`.

5.68.2.21 `template<typename _CharT, typename _Traits> virtual pos_type std::basic_streambuf<_CharT, _Traits>::seekoff (off_type, ios_base::seekdir, ios_base::openmode = ios_base::in | ios_base::out) [inline],`
`[protected], [virtual], [inherited]`

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented in [std::basic_filebuf<_CharT, _Traits>](#), [std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>](#), [std::basic_filebuf<char_type, traits_type>](#), and [std::basic_stringbuf<_CharT, _Traits, _Alloc>](#).

Definition at line 602 of file `streambuf`.

5.68.2.22 `template<typename _CharT, typename _Traits> virtual pos_type std::basic_streambuf<_CharT, _Traits>::seekpos (pos_type, ios_base::openmode = ios_base::in | ios_base::out) [inline],`
`[protected], [virtual], [inherited]`

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented in [std::basic_filebuf<_CharT, _Traits>](#), [std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>](#), [std::basic_filebuf<char_type, traits_type>](#), and [std::basic_stringbuf<_CharT, _Traits, _Alloc>](#).

Definition at line 614 of file `streambuf`.

5.68.2.23 `template<typename _CharT, typename _Traits> virtual basic_streambuf<char_type, Traits>*`
`std::basic_streambuf< _CharT, _Traits >::setbuf (char_type *, streamsize)` `[inline]`,
`[protected]`, `[virtual]`, `[inherited]`

Manipulates the buffer.

Each derived class provides its own appropriate behavior. See the next-to-last paragraph of <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html#io.streambuf.buffering> for more on this function.

Note

Base class version does nothing, returns `this`.

Reimplemented in `std::basic_filebuf< _CharT, _Traits >`, `std::basic_filebuf< _CharT, encoding_char_traits< _CharT >`, `std::basic_filebuf< char_type, traits_type >`, and `std::basic_stringbuf< _CharT, _Traits, _Alloc >`.

Definition at line 591 of file `streambuf`.

5.68.2.24 `template<typename _CharT, typename _Traits> void std::basic_streambuf< _CharT, _Traits >::setg (char_type *`
`__gbeg, char_type * __gnext, char_type * __gend)` `[inline]`, `[protected]`, `[inherited]`

Setting the three read area pointers.

Parameters

<code>__gbeg</code>	A pointer.
<code>__gnext</code>	A pointer.
<code>__gend</code>	A pointer.

Postcondition

`__gbeg == eback()`, `__gnext == gptr()`, and `__gend == egptr()`

Definition at line 509 of file `streambuf`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

5.68.2.25 `template<typename _CharT, typename _Traits> void std::basic_streambuf< _CharT, _Traits >::setp (char_type *`
`__pbeg, char_type * __pend)` `[inline]`, `[protected]`, `[inherited]`

Setting the three write area pointers.

Parameters

<code>__pbeg</code>	A pointer.
<code>__pend</code>	A pointer.

Postcondition

`__pbeg == pbase()`, `__pbeg == pptr()`, and `__pend == epptr()`

Definition at line 555 of file `streambuf`.

5.68.2.26 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::sgetc ()`
`[inline], [inherited]`

Getting the next character.

Returns

The next character, or eof.

If the input read position is available, returns that character, otherwise calls and returns `underflow()`. Does not move the read position after fetching the character.

Definition at line 338 of file `streambuf`.

Referenced by `std::istreambuf_iterator<_CharT, _Traits>::equal()`, `std::ostreambuf_iterator<_CharT, _Traits>::failed()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, and `std::basic_streambuf<_CharT, _Traits>::xsputn()`.

5.68.2.27 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf<_CharT, _Traits>::sgetn (`
`char_type * __s, streamsize __n) [inline], [inherited]`

Entry point for `xsgetn`.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	A count.

Returns `xsgetn(__s, __n)`. The effect is to fill `__s[0]` through `__s[__n-1]` with characters from the input sequence, if possible.

Definition at line 357 of file `streambuf`.

5.68.2.28 `template<typename _CharT, typename _Traits> virtual streamsize std::basic_streambuf<_CharT, _Traits>::showmanyc ()`
`[inline], [protected], [virtual], [inherited]`

Investigating the data available.

Returns

An estimate of the number of characters available in the input sequence, or -1.

If it returns a positive value, then successive calls to `underflow()` will not return `traits::eof()` until at least that number of characters have been supplied. If `showmanyc()` returns -1, then calls to `underflow()` or `uflow()` will fail. [27.5.2.4.3]/1

Note

Base class version does nothing, returns zero.

The standard adds that *the intention is not only that the calls [to `underflow` or `uflow`] will not return `eof()` but that they will return immediately.*

The standard adds that *the morphemes of `showmanyc` are **es-how-many-see**, not **show-manic**.*

Reimplemented in `std::basic_filebuf<_CharT, _Traits>`, `std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>`, `std::basic_filebuf<char_type, traits_type>`, and `std::basic_stringbuf<_CharT, _Traits, _Alloc>`.

Definition at line 649 of file `streambuf`.

```
5.68.2.29 template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::snextc ( )
        [inline], [inherited]
```

Getting the next character.

Returns

The next character, or `eof`.

Calls `sbumpc()`, and if that function returns `traits::eof()`, so does this function. Otherwise, `sgetc()`.

Definition at line 298 of file `streambuf`.

Referenced by `std::ostreambuf_iterator<_CharT, _Traits>::failed()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, and `std::basic_streambuf<_CharT, _Traits>::xsputn()`.

```
5.68.2.30 template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::sputbackc (
        char_type __c ) [inline], [inherited]
```

Pushing characters back into the input stream.

Parameters

<code>__c</code>	The character to push back.
------------------	-----------------------------

Returns

The previous character, if possible.

Similar to `sungetc()`, but `__c` is pushed onto the stream instead of *the previous character*. If successful, the next character fetched from the input stream will be `__c`.

Definition at line 372 of file `streambuf`.

Referenced by `std::basic_istream<_CharT, _Traits>::putback()`.

5.68.2.31 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::sputc (char_type __c) [inline], [inherited]`

Entry point for all single-character output functions.

Parameters

<code>__c</code>	A character to output.
------------------	------------------------

Returns

`__c`, if possible.

One of two public output functions.

If a write position is available for the output sequence (i.e., the buffer is not full), stores `__c` in that position, increments the position, and returns `traits::to_int_type(__c)`. If a write position is not available, returns `overflow(__c)`.

Definition at line 424 of file `streambuf`.

Referenced by `std::basic_istream<_CharT, _Traits>::get()`, `std::ostreambuf_iterator<_CharT, _Traits>::operator=()`, and `std::basic_streambuf<_CharT, _Traits>::xsputn()`.

5.68.2.32 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf<_CharT, _Traits>::sputn (const char_type * __s, streamsize __n) [inline], [inherited]`

Entry point for all single-character output functions.

Parameters

<code>__s</code>	A buffer read area.
<code>__n</code>	A count.

One of two public output functions.

Returns `xspn(__s,__n)`. The effect is to write `__s[0]` through `__s[__n-1]` to the output sequence, if possible.

Definition at line 450 of file `streambuf`.

Referenced by `std::ostreambuf_iterator<_CharT,_Traits>::failed()`.

5.68.2.33 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT,_Traits>::sungetc ()`
`[inline],[inherited]`

Moving backwards in the input stream.

Returns

The previous character, if possible.

If a putback position is available, this function decrements the input pointer and returns that character. Otherwise, calls and returns `pbackfail()`. The effect is to *unget* the last character *gotten*.

Definition at line 397 of file `streambuf`.

Referenced by `std::basic_istream<_CharT,_Traits>::unget()`.

5.68.2.34 `template<typename _CharT, typename _Traits = std::char_traits<_CharT>> virtual int __gnu_↵`
`cxx::stdio_sync_filebuf<_CharT,_Traits>::sync (void) [inline],[protected],`
`[virtual]`

Synchronizes the buffer arrays with the controlled sequences.

Returns

-1 on failure.

Each derived class provides its own appropriate behavior, including the definition of *failure*.

Note

Base class version does nothing, returns zero.

Reimplemented from [std::basic_streambuf<_CharT,_Traits>](#).

Definition at line 190 of file `stdio_sync_filebuf.h`.

References `std::ios_base::beg`, `std::ios_base::cur`, `std::ios_base::in`, and `std::ios_base::out`.

5.68.2.35 `template<typename _CharT, typename _Traits = std::char_traits<_CharT>> virtual int_type
__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>::uflow () [inline], [protected], [virtual]`

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function does the same thing as `underflow()`, and in fact is required to call that function. It also returns the new character, like `underflow()` does. However, this function also moves the read position forward by one.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

Definition at line 138 of file `stdio_sync_filebuf.h`.

5.68.2.36 `template<typename _CharT, typename _Traits = std::char_traits<_CharT>> virtual int_type
__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>::underflow () [inline], [protected],
[virtual]`

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function is called when the input buffer is exhausted (or does not exist, as buffering need not actually be done). If a buffer exists, it is *refilled*. In either case, the next available character is returned, or `traits::eof()` to indicate a null pending sequence.

For a formal definition of the pending sequence, see a good text such as Langer & Kreft, or [27.5.2.4.3]/7-14.

A functioning input streambuf can be created by overriding only this function (no buffer area will be used). For an example, see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html>

Note

Base class version does nothing, returns `eof()`.



Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

Definition at line 131 of file `stdio_sync_filebuf.h`.

5.68.2.37 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf<_CharT, _Traits>::xsgetn (
char_type * __s, streamsize __n) [protected], [virtual], [inherited]`

Multiple character extraction.

Parameters

 <code>__s</code>	A buffer area.
 <code>__n</code>	Maximum number of characters to assign.

Returns

The number of characters assigned.

Fills `__s[0]` through `__s[__n-1]` with characters from the input sequence, as if by `sbumpc()`. Stops when either `__n` characters have been copied, or when `traits::eof()` would be copied.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented in [std::basic_filebuf<_CharT, _Traits>](#), [std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>](#), and [std::basic_filebuf<char_type, traits_type>](#).

Definition at line 46 of file `streambuf.tcc`.



References `std::min()`, and `std::basic_streambuf<_CharT, _Traits>::xspn()`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::xsgetn()`.

5.68.2.38 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf<_CharT, _Traits>::xspn (const char_type * __s, streamsize __n) [protected], [virtual], [inherited]`

Multiple character insertion.

Parameters

 <code>__s</code>	A buffer area.
 <code>__n</code>	Maximum number of characters to write.

Returns

The number of characters written.

Writes `__s[0]` through `__s[__n-1]` to the output sequence, as if by `sputc()`. Stops when either `n` characters have been copied, or when `sputc()` would return `traits::eof()`.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented in [std::basic_filebuf<_CharT, _Traits>](#), [std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>](#), and [std::basic_filebuf<char_type, traits_type>](#).

Definition at line 80 of file `streambuf.tcc`.

References `std::min()`, `std::basic_streambuf<_CharT, _Traits>::sgetc()`, `std::basic_streambuf<_CharT, _Traits>::snnextc()`, and `std::basic_streambuf<_CharT, _Traits>::sputc()`.

Referenced by `std::basic_streambuf<_CharT, _Traits>::xsgetn()`, and `std::basic_filebuf<_CharT, _Traits>::xsputn()`.

5.68.3 Member Data Documentation

5.68.3.1 `template<typename _CharT, typename _Traits> locale std::basic_streambuf<_CharT, _Traits>::_M_buf_locale`
[protected], [inherited]

Current locale setting.

Definition at line 192 of file `streambuf`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::basic_filebuf()`.

5.68.3.2 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::_M_in_beg`
[protected], [inherited]

Start of get area.

Definition at line 184 of file `streambuf`.

5.68.3.3 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::_M_in_cur`
[protected], [inherited]

Current read area.

Definition at line 185 of file `streambuf`.

5.68.3.4 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::_M_in_end`
[protected], [inherited]

End of get area.

Definition at line 186 of file `streambuf`.

5.68.3.5 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::_M_out_beg`
[protected], [inherited]

Start of put area.

Definition at line 187 of file `streambuf`.

5.68.3.6 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_cur`
`[protected], [inherited]`

Current put area.

Definition at line 188 of file streambuf.

5.68.3.7 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_end`
`[protected], [inherited]`

End of put area.

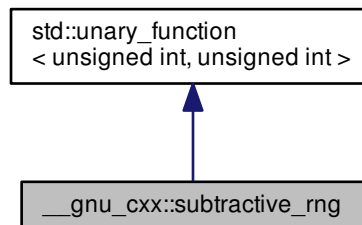
Definition at line 189 of file streambuf.

The documentation for this class was generated from the following file:

- [stdio_sync_filebuf.h](#)

5.69 `__gnu_cxx::subtractive_rng` Class Reference

Inheritance diagram for `__gnu_cxx::subtractive_rng`:



Public Types

- `typedef _Arg` [argument_type](#)
- `typedef _Result` [result_type](#)

Public Member Functions

- [subtractive_rng](#) (unsigned int __seed)
- [subtractive_rng](#) ()
- `void` [_M_initialize](#) (unsigned int __seed)
- unsigned int [operator\(\)](#) (unsigned int __limit)

5.69.1 Detailed Description

The `subtractive_rng` class is documented on [SGI's site](#). Note that this code assumes that `int` is 32 bits.

Definition at line 352 of file `ext/functional`.

5.69.2 Member Typedef Documentation

5.69.2.1 `template<typename _Arg, typename _Result> typedef _Arg std::unary_function< _Arg, _Result >::argument_type`
[inherited]

`argument_type` is the type of the argument

Definition at line 108 of file `stl_function.h`.

5.69.2.2 `template<typename _Arg, typename _Result> typedef _Result std::unary_function< _Arg, _Result >::result_type`
[inherited]

`result_type` is the return type

Definition at line 111 of file `stl_function.h`.

5.69.3 Constructor & Destructor Documentation

5.69.3.1 `__gnu_cxx::subtractive_rng::subtractive_rng (unsigned int __seed)` [inline]

Ctor allowing you to initialize the seed.

Definition at line 394 of file `ext/functional`.

5.69.3.2 `__gnu_cxx::subtractive_rng::subtractive_rng ()` [inline]

Default ctor; initializes its state with some number you don't see.

Definition at line 398 of file `ext/functional`.

5.69.4 Member Function Documentation

5.69.4.1 `unsigned int __gnu_cxx::subtractive_rng::operator()(unsigned int __limit)` [inline]

Returns a number less than the argument.

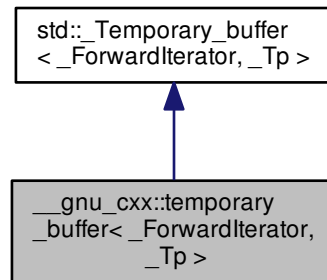
Definition at line 363 of file `ext/functional`.

The documentation for this class was generated from the following file:

- [ext/functional](#)

5.70 `__gnu_cxx::temporary_buffer<_ForwardIterator, _Tp>` Struct Template Reference

Inheritance diagram for `__gnu_cxx::temporary_buffer<_ForwardIterator, _Tp>`:



Public Types

- typedef pointer **iterator**
- typedef value_type * **pointer**
- typedef ptrdiff_t **size_type**
- typedef _Tp **value_type**

Public Member Functions

- [temporary_buffer](#) (`_ForwardIterator __first, _ForwardIterator __last`)
- [~temporary_buffer](#) ()
- iterator [begin](#) ()
- iterator [end](#) ()
- size_type [requested_size](#) () const
- size_type [size](#) () const

Protected Attributes

- pointer **_M_buffer**
- size_type **_M_len**
- size_type **_M_original_len**

5.70.1 Detailed Description

```
template<class _ForwardIterator, class _Tp = typename std::iterator_traits<_ForwardIterator>::value_type>
struct __gnu_cxx::temporary_buffer<_ForwardIterator, _Tp>
```

This class provides similar behavior and semantics of the standard functions `get_temporary_buffer()` and `return_temporary_buffer()`, but encapsulated in a type vaguely resembling a standard container.

By default, a `temporary_buffer<Iter>` stores space for objects of whatever type the `Iter` iterator points to. It is constructed from a typical `[first,last)` range, and provides the `begin()`, `end()`, `size()` functions, as well as `requested_size()`. For non-trivial types, copies of `*first` will be used to initialize the storage.

`malloc` is used to obtain underlying storage.

Like `get_temporary_buffer()`, not all the requested memory may be available. Ideally, the created buffer will be large enough to hold a copy of `[first,last)`, but if `size()` is less than `requested_size()`, then this didn't happen.

Definition at line 183 of file `ext/memory`.

5.70.2 Constructor & Destructor Documentation

```
5.70.2.1 template<class _ForwardIterator, class _Tp = typename std::iterator_traits<_ForwardIterator>::value_type>
        __gnu_cxx::temporary_buffer<_ForwardIterator, _Tp>::temporary_buffer ( _ForwardIterator __first,
        _ForwardIterator __last ) [inline]
```

Requests storage large enough to hold a copy of `[first,last)`.

Definition at line 186 of file `ext/memory`.

```
5.70.2.2 template<class _ForwardIterator, class _Tp = typename std::iterator_traits<_ForwardIterator>::value_type>
        __gnu_cxx::temporary_buffer<_ForwardIterator, _Tp>::~~temporary_buffer ( ) [inline]
```

Destroys objects and frees storage.

Definition at line 190 of file `ext/memory`.

5.70.3 Member Function Documentation

```
5.70.3.1 template<typename _ForwardIterator, typename _Tp> iterator std::_Temporary_buffer<_ForwardIterator, _Tp>::begin ( ) [inline], [inherited]
```

As per Table mumble.

Definition at line 151 of file `stl_tempbuf.h`.

Referenced by `std::__stable_partition_adaptive()`.

5.70.3.2 `template<typename _ForwardIterator, typename _Tp> iterator std::_Temporary_buffer< _ForwardIterator, _Tp >::end () [inline],[inherited]`

As per Table mumble.

Definition at line 156 of file `stl_tempbuf.h`.

References `std::_addressof()`, `std::_Construct()`, `std::_Destroy()`, `std::_Temporary_buffer< _ForwardIterator, _Tp >::_Temporary_buffer()`, and `std::return_temporary_buffer()`.

5.70.3.3 `template<typename _ForwardIterator, typename _Tp> size_type std::_Temporary_buffer< _ForwardIterator, _Tp >::requested_size () const [inline],[inherited]`

Returns the size requested by the constructor; may be `>size()`.

Definition at line 146 of file `stl_tempbuf.h`.

Referenced by `std::__stable_partition_adaptive()`.

5.70.3.4 `template<typename _ForwardIterator, typename _Tp> size_type std::_Temporary_buffer< _ForwardIterator, _Tp >::size () const [inline],[inherited]`

As per Table mumble.

Definition at line 141 of file `stl_tempbuf.h`.

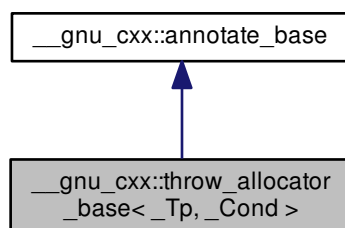
Referenced by `std::__stable_partition_adaptive()`.

The documentation for this struct was generated from the following file:

- [ext/memory](#)

5.71 `__gnu_cxx::throw_allocator_base< _Tp, _Cond >` Class Template Reference

Inheritance diagram for `__gnu_cxx::throw_allocator_base< _Tp, _Cond >`:



Public Types

- typedef const value_type * **const_pointer**
- typedef const value_type & **const_reference**
- typedef ptrdiff_t **difference_type**
- typedef value_type * **pointer**
- typedef [std::true_type](#) **propagate_on_container_move_assignment**
- typedef value_type & **reference**
- typedef size_t **size_type**
- typedef _Tp **value_type**

Public Member Functions

- pointer **address** (reference __x) const noexcept
- const_pointer **address** (const_reference __x) const noexcept
- pointer **allocate** (size_type __n, [std::allocator](#)< void >::const_pointer hint=0)
- void **check** (size_type __n)
- void **check_allocated** (void *p, size_t size)
- void **check_allocated** (pointer __p, size_type __n)
- void **check_constructed** (void *p)
- void **check_constructed** (size_t label)
- template<typename _Up, typename... _Args>
void **construct** (_Up *__p, _Args &&... __args)
- void **deallocate** (pointer __p, size_type __n)
- template<typename _Up >
void **destroy** (_Up *__p)
- void **erase** (void *p, size_t size)
- void **erase_construct** (void *p)
- void **insert** (void *p, size_t size)
- void **insert_construct** (void *p)
- size_type **max_size** () const noexcept

Static Public Member Functions

- static void **check** ()
- static size_t **get_label** ()
- static void **set_label** (size_t l)

5.71.1 Detailed Description

```
template<typename _Tp, typename _Cond>
class __gnu_cxx::throw_allocator_base<_Tp, _Cond>
```

Allocator class with logging and exception generation control. Intended to be used as an allocator_type in templated code.

Note: Deallocate not allowed to throw.

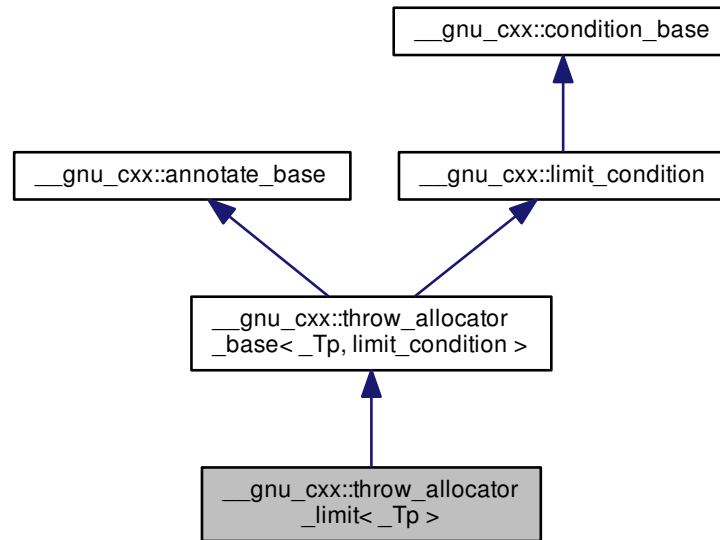
Definition at line 790 of file throw_allocator.h.

The documentation for this class was generated from the following file:

- [throw_allocator.h](#)

5.72 `__gnu_cxx::throw_allocator_limit<_Tp>` Struct Template Reference

Inheritance diagram for `__gnu_cxx::throw_allocator_limit<_Tp>`:



Public Types

- typedef const value_type * **const_pointer**
- typedef const value_type & **const_reference**
- typedef ptrdiff_t **difference_type**
- typedef value_type * **pointer**
- typedef [std::true_type](#) **propagate_on_container_move_assignment**
- typedef value_type & **reference**
- typedef size_t **size_type**
- typedef _Tp **value_type**

Public Member Functions

- **throw_allocator_limit** (const [throw_allocator_limit](#) &) noexcept
- template<typename _Tp1 >
 throw_allocator_limit (const [throw_allocator_limit](#)<_Tp1> &) noexcept
- pointer **address** (reference __x) const noexcept
- const_pointer **address** (const_reference __x) const noexcept
- pointer **allocate** (size_type __n, [std::allocator](#)< void >::const_pointer hint=0)
- void **check** (size_type __n)
- void **check_allocated** (void *p, size_t size)

- void **check_allocated** (pointer __p, size_type __n)
- void **check_constructed** (void *p)
- void **check_constructed** (size_t label)
- void **construct** (_Up *__p, _Args &&... __args)
- void **deallocate** (pointer __p, size_type __n)
- void **destroy** (_Up *__p)
- void **erase** (void *p, size_t size)
- void **erase_construct** (void *p)
- void **insert** (void *p, size_t size)
- void **insert_construct** (void *p)
- size_type **max_size** () const noexcept

Static Public Member Functions

- static void **check** ()
- static size_t & **count** ()
- static size_t **get_label** ()
- static size_t & **limit** ()
- static void **set_label** (size_t l)
- static void **set_limit** (const size_t __l)
- static void **throw_conditionally** ()

5.72.1 Detailed Description

```
template<typename _Tp>
struct __gnu_cxx::throw_allocator_limit<_Tp>
```

Allocator throwing via limit condition.

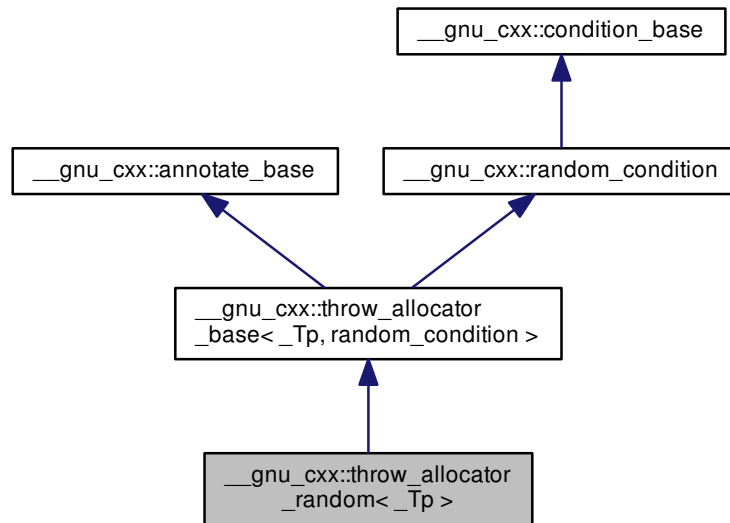
Definition at line 899 of file `throw_allocator.h`.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

5.73 `__gnu_cxx::throw_allocator_random<_Tp>` Struct Template Reference

Inheritance diagram for `__gnu_cxx::throw_allocator_random<_Tp>`:



Public Types

- typedef const value_type * **const_pointer**
- typedef const value_type & **const_reference**
- typedef ptrdiff_t **difference_type**
- typedef value_type * **pointer**
- typedef [std::true_type](#) **propagate_on_container_move_assignment**
- typedef value_type & **reference**
- typedef size_t **size_type**
- typedef _Tp **value_type**

Public Member Functions

- **throw_allocator_random** (const [throw_allocator_random](#) &) noexcept
- template<typename _Tp1 >
 throw_allocator_random (const [throw_allocator_random](#)<_Tp1 > &) noexcept
- pointer **address** (reference __x) const noexcept
- const_pointer **address** (const_reference __x) const noexcept
- pointer **allocate** (size_type __n, [std::allocator](#)< void >::const_pointer hint=0)
- void **check** (size_type __n)
- void **check_allocated** (void *p, size_t size)
- void **check_allocated** (pointer __p, size_type __n)

- void **check_constructed** (void *p)
- void **check_constructed** (size_t label)
- void **construct** (_Up *__p, _Args &&... __args)
- void **deallocate** (pointer __p, size_type __n)
- void **destroy** (_Up *__p)
- void **erase** (void *p, size_t size)
- void **erase_construct** (void *p)
- void **insert** (void *p, size_t size)
- void **insert_construct** (void *p)
- size_type **max_size** () const noexcept
- void **seed** (unsigned long __s)

Static Public Member Functions

- static void **check** ()
- static size_t **get_label** ()
- static void **set_label** (size_t l)
- static void **set_probability** (double __p)
- static void **throw_conditionally** ()

5.73.1 Detailed Description

```
template<typename _Tp>
struct __gnu_cxx::throw_allocator_random<_Tp>
```

Allocator throwing via random condition.

Definition at line 920 of file `throw_allocator.h`.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

5.74 `__gnu_cxx::throw_value_base<_Cond>` Struct Template Reference

Inherits `_Cond`.

Public Types

- typedef `_Cond` **condition_type**

Public Member Functions

- **throw_value_base** (const [throw_value_base](#) &__v)
- **throw_value_base** ([throw_value_base](#) &&)=default
- **throw_value_base** (const std::size_t __i)
- [throw_value_base](#) & **operator++** ()
- [throw_value_base](#) & **operator=** (const [throw_value_base](#) &__v)
- [throw_value_base](#) & **operator=** ([throw_value_base](#) &&)=default

Public Attributes

- std::size_t **M_i**

5.74.1 Detailed Description

```
template<typename _Cond>
struct __gnu_cxx::throw_value_base< _Cond >
```

Class with exception generation control. Intended to be used as a value_type in templated code.

Note: Destructor not allowed to throw.

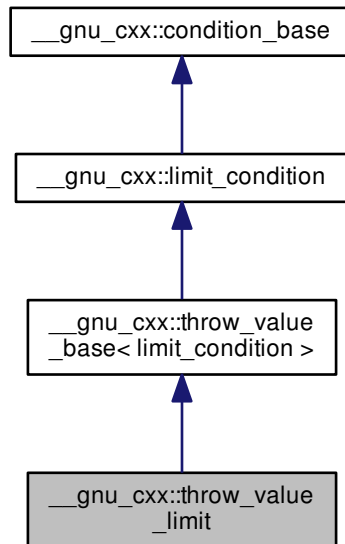
Definition at line 603 of file throw_allocator.h.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

5.75 `__gnu_cxx::throw_value_limit` Struct Reference

Inheritance diagram for `__gnu_cxx::throw_value_limit`:



Public Types

- typedef `throw_value_base< limit_condition >` **base_type**
- typedef `limit_condition` **condition_type**

Public Member Functions

- **throw_value_limit** (const `throw_value_limit` &__other)
- **throw_value_limit** (`throw_value_limit` &&)=default
- **throw_value_limit** (const std::size_t __i)
- `throw_value_base` & **operator++** ()
- `throw_value_limit` & **operator=** (const `throw_value_limit` &__other)
- `throw_value_limit` & **operator=** (`throw_value_limit` &&)=default

Static Public Member Functions

- static size_t & **count** ()
- static size_t & **limit** ()
- static void **set_limit** (const size_t __i)
- static void **throw_conditionally** ()

Public Attributes

- `std::size_t _M_i`

5.75.1 Detailed Description

Type throwing via limit condition.

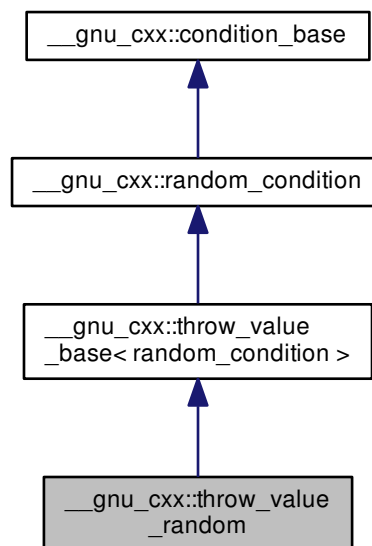
Definition at line 720 of file `throw_allocator.h`.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

5.76 `__gnu_cxx::throw_value_random` Struct Reference

Inheritance diagram for `__gnu_cxx::throw_value_random`:



Public Types

- typedef `throw_value_base< random_condition >` **base_type**
- typedef `random_condition` **condition_type**

Public Member Functions

- `throw_value_random` (const [throw_value_random](#) &__other)
- `throw_value_random` ([throw_value_random](#) &&)=default
- `throw_value_random` (const std::size_t __i)
- [throw_value_base](#) & `operator++` ()
- `throw_value_random` & `operator=` (const [throw_value_random](#) &__other)
- `throw_value_random` & `operator=` ([throw_value_random](#) &&)=default
- void `seed` (unsigned long __s)

Static Public Member Functions

- static void `set_probability` (double __p)
- static void `throw_conditionally` ()

Public Attributes

- std::size_t `_M_i`

5.76.1 Detailed Description

Type throwing via random condition.

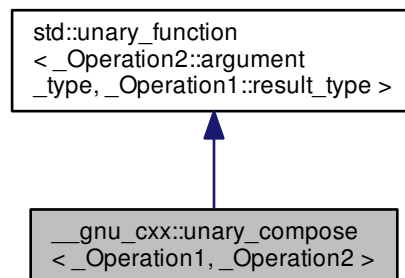
Definition at line 751 of file `throw_allocator.h`.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

5.77 `__gnu_cxx::unary_compose< _Operation1, _Operation2 >` Class Template Reference

Inheritance diagram for `__gnu_cxx::unary_compose< _Operation1, _Operation2 >`:



Public Types

- typedef `_Arg` [argument_type](#)
- typedef `_Result` [result_type](#)

Public Member Functions

- **unary_compose** (const `_Operation1` &__x, const `_Operation2` &__y)
- `_Operation1::result_type` **operator()** (const typename `_Operation2::argument_type` &__x) const

Protected Attributes

- `_Operation1` **_M_fn1**
- `_Operation2` **_M_fn2**

5.77.1 Detailed Description

```
template<class _Operation1, class _Operation2>
class __gnu_cxx::unary_compose< _Operation1, _Operation2 >
```

An [SGI extension](#) .

Definition at line 125 of file `ext/functional`.

5.77.2 Member Typedef Documentation

5.77.2.1 `template<typename _Arg, typename _Result> typedef _Arg std::unary_function< _Arg, _Result >::argument_type`
[*inherited*]

`argument_type` is the type of the argument

Definition at line 108 of file `stl_function.h`.

5.77.2.2 `template<typename _Arg, typename _Result> typedef _Result std::unary_function< _Arg, _Result >::result_type`
[*inherited*]

`result_type` is the return type

Definition at line 111 of file `stl_function.h`.

The documentation for this class was generated from the following file:

- [ext/functional](#)

5.78 `__gnu_debug::After_nth_from<_Iterator>` Class Template Reference

Public Member Functions

- **After_nth_from** (const difference_type &__n, const _Iterator &__base)
- bool **operator()** (const _Iterator &__x) const

5.78.1 Detailed Description

```
template<typename _Iterator>
class __gnu_debug::After_nth_from<_Iterator>
```

A function object that returns true when the given random access iterator is at least `n` steps away from the given iterator.

Definition at line 74 of file `safe_sequence.h`.

The documentation for this class was generated from the following file:

- [safe_sequence.h](#)

5.79 `__gnu_debug::BeforeBeginHelper<_Sequence>` Struct Template Reference

Static Public Member Functions

- template<typename _Iterator>
static bool **_S_Is** (const [_Safe_iterator](#)<_Iterator, _Sequence> &)
- template<typename _Iterator>
static bool **_S_Is_Beginnest** (const [_Safe_iterator](#)<_Iterator, _Sequence> &__it)

5.79.1 Detailed Description

```
template<typename _Sequence>
struct __gnu_debug::BeforeBeginHelper<_Sequence>
```

Helper struct to deal with sequence offering a `before_begin` iterator.

Definition at line 45 of file `safe_iterator.h`.

The documentation for this struct was generated from the following file:

- [safe_iterator.h](#)

5.80 `__gnu_debug::Equal_to<_Type>` Class Template Reference

Public Member Functions

- `_Equal_to` (const `_Type` &__v)
- `bool operator()` (const `_Type` &__x) const

5.80.1 Detailed Description

```
template<typename _Type>
class __gnu_debug::Equal_to<_Type>
```

A simple function object that returns true if the passed-in value is equal to the stored value.

Definition at line 59 of file `safe_sequence.h`.

The documentation for this class was generated from the following file:

- [safe_sequence.h](#)

5.81 `__gnu_debug::Not_equal_to<_Type>` Class Template Reference

Public Member Functions

- `_Not_equal_to` (const `_Type` &__v)
- `bool operator()` (const `_Type` &__x) const

5.81.1 Detailed Description

```
template<typename _Type>
class __gnu_debug::Not_equal_to<_Type>
```

A simple function object that returns true if the passed-in value is not equal to the stored value. It saves typing over using both `bind1st` and `not_equal`.

Definition at line 44 of file `safe_sequence.h`.

The documentation for this class was generated from the following file:

- [safe_sequence.h](#)

5.82 `__gnu_debug::Safe_container<_SafeContainer, _Alloc, _SafeBase, _IsCxx11AllocatorAware>` Class Template Reference

Inherits `_SafeBase<_SafeContainer>`.

Public Member Functions

- void **_M_swap** ([_Safe_container](#) &__x) noexcept
- [_Safe_container](#) & **operator=** (const [_Safe_container](#) &) noexcept
- [_Safe_container](#) & **operator=** ([_Safe_container](#) &&__x) noexcept

Protected Member Functions

- **_Safe_container** (const [_Safe_container](#) &)=default
- **_Safe_container** ([_Safe_container](#) &&__x) noexcept
- **_Safe_container** ([_Safe_container](#) &&__x, const [_Alloc](#) &__a)
- [_Safe_container](#) & **_M_safe** () noexcept

5.82.1 Detailed Description

```
template<typename _SafeContainer, typename _Alloc, template< typename > class _SafeBase, bool _IsCxx11AllocatorAware = true>
class __gnu_debug::_Safe_container< _SafeContainer, _Alloc, _SafeBase, _IsCxx11AllocatorAware >
```

Safe class dealing with some allocator dependent operations.

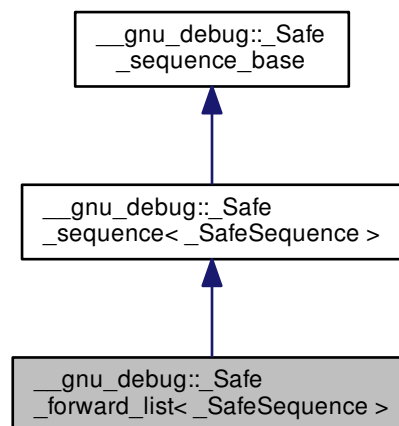
Definition at line 41 of file `safe_container.h`.

The documentation for this class was generated from the following file:

- [safe_container.h](#)

5.83 __gnu_debug::_Safe_forward_list<_SafeSequence> Class Template Reference

Inheritance diagram for `__gnu_debug::_Safe_forward_list<_SafeSequence>`:



Public Member Functions

- void `_M_attach` (`_Safe_iterator_base` * __it, bool __constant)
- void `_M_attach_single` (`_Safe_iterator_base` * __it, bool __constant) throw ()
- void `_M_detach` (`_Safe_iterator_base` * __it)
- void `_M_detach_single` (`_Safe_iterator_base` * __it) throw ()
- void `_M_invalidate_all` () const
- void `_M_invalidate_if` (`_Predicate` __pred)
- void `_M_transfer_from_if` (`_Safe_sequence` & __from, `_Predicate` __pred)

Public Attributes

- `_Safe_iterator_base` * `_M_const_iterators`
- `_Safe_iterator_base` * `_M_iterators`
- unsigned int `_M_version`

Protected Member Functions

- void `_M_detach_all` ()
- void `_M_detach_singular` ()
- `__gnu_cxx::__mutex` & `_M_get_mutex` () throw ()
- void `_M_invalidate_all` ()
- void `_M_revalidate_singular` ()
- void `_M_swap` (`_Safe_sequence_base` &) noexcept

5.83.1 Detailed Description

```
template<typename _SafeSequence>
class __gnu_debug::_Safe_forward_list< _SafeSequence >
```

Special iterators swap and invalidation for `forward_list` because of the `before_begin` iterator.

Definition at line 51 of file `debug/forward_list`.

5.83.2 Member Function Documentation

5.83.2.1 void `__gnu_debug::_Safe_sequence_base::_M_attach` (`_Safe_iterator_base` * __it, bool __constant)
[inherited]

Attach an iterator to this sequence.

5.83.2.2 void `__gnu_debug::_Safe_sequence_base::_M_attach_single` (`_Safe_iterator_base` * __it, bool __constant) throw ()
[inherited]

Likewise but not thread safe.

5.83.2.3 `void __gnu_debug::Safe_sequence_base::M_detach (_Safe_iterator_base * __it)` [inherited]

Detach an iterator from this sequence

Referenced by `__gnu_debug::Safe_iterator<_Iterator, _Sequence>::__Safe_iterator()`.

5.83.2.4 `void __gnu_debug::Safe_sequence_base::M_detach_all ()` [protected],[inherited]

Detach all iterators, leaving them singular.

5.83.2.5 `void __gnu_debug::Safe_sequence_base::M_detach_single (_Safe_iterator_base * __it) throw`
[inherited]

Likewise but not thread safe.

5.83.2.6 `void __gnu_debug::Safe_sequence_base::M_detach_singular ()` [protected],[inherited]

Detach all singular iterators.

Postcondition

for all iterators *i* attached to this sequence, *i*->_M_version == _M_version.

5.83.2.7 `__gnu_cxx::mutex& __gnu_debug::Safe_sequence_base::M_get_mutex () throw` [protected],
[inherited]

For use in `_Safe_sequence`.

Referenced by `__gnu_debug::Safe_sequence<_Sequence>::__M_transfer_from_if()`.

5.83.2.8 `void __gnu_debug::Safe_sequence_base::M_invalidate_all () const` [inline],[inherited]

Invalidates all iterators.

Definition at line 248 of file `safe_base.h`.

References `__gnu_debug::Safe_iterator_base::M_attach()`, `__gnu_debug::Safe_iterator_base::M_attach_single()`, `__gnu_debug::Safe_iterator_base::M_detach()`, and `__gnu_debug::Safe_iterator_base::M_detach_single()`.

5.83.2.9 `void __gnu_debug::Safe_sequence<_SafeSequence>::__M_invalidate_if (_Predicate __pred)`
[inherited]

Invalidates all iterators *x* that reference this sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

5.83.2.10 `void __gnu_debug::Safe_sequence_base::M_revalidate_singular ()` [protected],[inherited]

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

5.83.2.11 `void __gnu_debug::_Safe_sequence<_SafeSequence>::M_transfer_from_if (_Safe_sequence<_SafeSequence> & __from, _Predicate __pred)` [inherited]

Transfers all iterators `x` that reference `from` sequence, are not singular, and for which `__pred(x)` returns `true`. `__pred` will be invoked with the normal iterators nested in the safe ones.

5.83.3 Member Data Documentation

5.83.3.1 `_Safe_iterator_base* __gnu_debug::_Safe_sequence_base::M_const_iterators` [inherited]

The list of constant iterators that reference this container.

Definition at line 193 of file `safe_base.h`.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::M_transfer_from_if()`.

5.83.3.2 `_Safe_iterator_base* __gnu_debug::_Safe_sequence_base::M_iterators` [inherited]

The list of mutable iterators that reference this container.

Definition at line 190 of file `safe_base.h`.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::M_transfer_from_if()`.

5.83.3.3 `unsigned int __gnu_debug::_Safe_sequence_base::M_version` [mutable], [inherited]

The container version number. This number may never be 0.

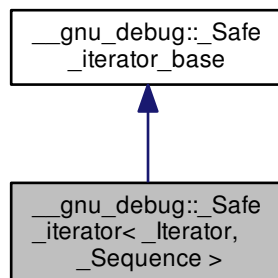
Definition at line 196 of file `safe_base.h`.

The documentation for this class was generated from the following file:

- [debug/forward_list](#)

5.84 `__gnu_debug::_Safe_iterator<_Iterator, _Sequence>` Class Template Reference

Inheritance diagram for `__gnu_debug::_Safe_iterator<_Iterator, _Sequence>`:



Public Types

- `typedef _Traits::difference_type` **difference_type**
- `typedef _Traits::iterator_category` **iterator_category**
- `typedef _Iterator` **iterator_type**
- `typedef _Traits::pointer` **pointer**
- `typedef _Traits::reference` **reference**
- `typedef _Traits::value_type` **value_type**

Public Member Functions

- `_Safe_iterator` () noexcept
- `_Safe_iterator` (const `_Iterator` &__i, const `_Safe_sequence_base` *__seq) noexcept
- `_Safe_iterator` (const `_Safe_iterator` &__x) noexcept
- `_Safe_iterator` (`_Safe_iterator` &&__x) noexcept
- `template<typename _MutableIterator >`
`_Safe_iterator` (const `_Safe_iterator`< `_MutableIterator`, `typename __gnu_cxx::enable_if<(std::is_same<_MutableIterator, typename _Sequence::iterator::iterator_type>>::value), _Sequence>::__type` &__x) noexcept
- `void` `_M_attach` (`_Safe_sequence_base` *__seq, bool __constant)
- `void` `_M_attach` (`_Safe_sequence_base` *__seq)
- `void` `_M_attach_single` (`_Safe_sequence_base` *__seq, bool __constant) throw ()
- `void` `_M_attach_single` (`_Safe_sequence_base` *__seq)
- `bool` `_M_attached_to` (const `_Safe_sequence_base` *__seq) const
- `bool` `_M_before_dereferenceable` () const
- `bool` `_M_can_advance` (const `difference_type` &__n) const
- `bool` `_M_can_compare` (const `_Safe_iterator_base` &__x) const throw ()
- `bool` `_M_decrementable` () const
- `bool` `_M_dereferenceable` () const
- `void` `_M_detach` ()
- `void` `_M_detach_single` () throw ()
- `__gnu_cxx::conditional_type< std::is_same<_Const_iterator, _Safe_iterator>::__value, const _Const_iterator, _Safe_iterator>::__type` `_M_get_sequence` () const
- `bool` `_M_incrementable` () const
- `void` `_M_invalidate` ()
- `bool` `_M_is_before_begin` () const
- `bool` `_M_is_begin` () const
- `bool` `_M_is_beginnest` () const
- `bool` `_M_is_end` () const
- `void` `_M_reset` () throw ()
- `bool` `_M_singular` () const throw ()
- `void` `_M_unlink` () throw ()
- `bool` `_M_valid_range` (const `_Safe_iterator` &__rhs, `std::pair< difference_type, _Distance_precision >` &__dist, bool __check_dereferenceable=true) const
- `_Iterator` & `base` () noexcept
- `const _Iterator` & `base` () const noexcept
- `operator _Iterator` () const noexcept
- `reference` `operator*` () const noexcept
- `_Safe_iterator` `operator+` (const `difference_type` &__n) const noexcept
- `_Safe_iterator` & `operator++` () noexcept

- `_Safe_iterator operator++` (int) noexcept
- `_Safe_iterator & operator+=` (const difference_type &__n) noexcept
- `_Safe_iterator operator-` (const difference_type &__n) const noexcept
- `_Safe_iterator & operator--` () noexcept
- `_Safe_iterator operator--` (int) noexcept
- `_Safe_iterator & operator-=` (const difference_type &__n) noexcept
- pointer `operator->` () const noexcept
- `_Safe_iterator & operator=` (const `_Safe_iterator` &__x) noexcept
- `_Safe_iterator & operator=` (`_Safe_iterator` &&__x) noexcept
- reference `operator[]` (const difference_type &__n) const noexcept

Public Attributes

- `_Safe_iterator_base * _M_next`
- `_Safe_iterator_base * _M_prior`
- `_Safe_sequence_base * _M_sequence`
- unsigned int `_M_version`

Protected Member Functions

- `__gnu_cxx::__mutex & _M_get_mutex` () throw ()

5.84.1 Detailed Description

```
template<typename _Iterator, typename _Sequence>
class __gnu_debug::_Safe_iterator< _Iterator, _Sequence >
```

Safe iterator wrapper.

The class template `_Safe_iterator` is a wrapper around an iterator that tracks the iterator's movement among sequences and checks that operations performed on the "safe" iterator are legal. In addition to the basic iterator operations (which are validated, and then passed to the underlying iterator), `_Safe_iterator` has member functions for iterator invalidation, attaching/detaching the iterator from sequences, and querying the iterator's state.

Note that `_Iterator` must be the first base class so that it gets initialized before the iterator is being attached to the container's list of iterators and it is being detached before `_Iterator` get destroyed. Otherwise it would result in a data race.

Definition at line 56 of file `formatter.h`.

5.84.2 Constructor & Destructor Documentation

```
5.84.2.1 template<typename _Iterator, typename _Sequence> __gnu_debug::_Safe_iterator< _Iterator, _Sequence
>::_Safe_iterator( ) [inline],[noexcept]
```

Postcondition

the iterator is singular and unattached

Definition at line 119 of file `safe_iterator.h`.

```
5.84.2.2 template<typename _Iterator, typename _Sequence> __gnu_debug::__Safe_iterator<_Iterator, _Sequence>
    >::__Safe_iterator( const _Iterator & __i, const _Safe_sequence_base * __seq ) [inline], [noexcept]
```

Safe iterator construction from an unsafe iterator and its sequence.

Precondition

`seq` is not NULL

Postcondition

this is not singular

Definition at line 128 of file `safe_iterator.h`.

```
5.84.2.3 template<typename _Iterator, typename _Sequence> __gnu_debug::__Safe_iterator<_Iterator, _Sequence>
    >::__Safe_iterator( const _Safe_iterator<_Iterator, _Sequence> & __x ) [inline], [noexcept]
```

Copy construction.

Definition at line 140 of file `safe_iterator.h`.

```
5.84.2.4 template<typename _Iterator, typename _Sequence> __gnu_debug::__Safe_iterator<_Iterator, _Sequence>
    >::__Safe_iterator( _Safe_iterator<_Iterator, _Sequence> && __x ) [inline], [noexcept]
```

Move construction.

Postcondition

`__x` is singular and unattached

Definition at line 158 of file `safe_iterator.h`.

References `__gnu_debug::__Safe_sequence_base::M_detach()`.

```
5.84.2.5 template<typename _Iterator, typename _Sequence> template<typename _MutableIterator >
    __gnu_debug::__Safe_iterator<_Iterator, _Sequence> >::__Safe_iterator( const _Safe_iterator<
    _MutableIterator, typename __gnu_cxx::__enable_if<(std::__are_same<_MutableIterator, typename
    _Sequence::iterator::iterator_type >::__value), _Sequence>::__type > & __x ) [inline], [noexcept]
```

Converting constructor from a mutable iterator to a constant iterator.

Definition at line 178 of file `safe_iterator.h`.

5.84.3 Member Function Documentation

5.84.3.1 `void __gnu_debug::Safe_iterator_base::M_attach (_Safe_sequence_base * __seq, bool __constant)`
`[inherited]`

Attaches this iterator to the given sequence, detaching it from whatever sequence it was attached to originally. If the new sequence is the NULL pointer, the iterator is left unattached.

Referenced by `__gnu_debug::Safe_sequence_base::M_invalidate_all()`, and `__gnu_debug::Safe_iterator_base::↵_Safe_iterator_base()`.

5.84.3.2 `template<typename _Iterator, typename _Sequence> void __gnu_debug::Safe_iterator< _Iterator, _Sequence >::M_attach (_Safe_sequence_base * __seq)` `[inline]`

Attach iterator to the given sequence.

Definition at line 417 of file `safe_iterator.h`.

5.84.3.3 `void __gnu_debug::Safe_iterator_base::M_attach_single (_Safe_sequence_base * __seq, bool __constant) throw)`
`[inherited]`

Likewise, but not thread-safe.

Referenced by `__gnu_debug::Safe_sequence_base::M_invalidate_all()`, and `__gnu_debug::Safe_iterator_base::↵_Safe_iterator_base()`.

5.84.3.4 `template<typename _Iterator, typename _Sequence> void __gnu_debug::Safe_iterator< _Iterator, _Sequence >::M_attach_single (_Safe_sequence_base * __seq)` `[inline]`

Likewise, but not thread-safe.

Definition at line 422 of file `safe_iterator.h`.

5.84.3.5 `bool __gnu_debug::Safe_iterator_base::M_attached_to (const _Safe_sequence_base * __seq) const`
`[inline], [inherited]`

Determines if we are attached to the given sequence.

Definition at line 129 of file `safe_base.h`.

References `__gnu_debug::Safe_iterator_base::M_can_compare()`, and `__gnu_debug::Safe_iterator_base::M_↵singular()`.

5.84.3.6 `template<typename _Iterator, typename _Sequence> bool __gnu_debug::Safe_iterator< _Iterator, _Sequence >::M_before_dereferenceable () const` `[inline]`

Is the iterator before a dereferenceable one?

Definition at line 432 of file `safe_iterator.h`.

5.84.3.7 `bool __gnu_debug::Safe_iterator_base::M_can_compare (const _Safe_iterator_base & __x) const throw)`
`[inherited]`

Can we compare this iterator to the given iterator `__x`? Returns true if both iterators are nonsingular and reference the same sequence.

Referenced by `__gnu_debug::Safe_iterator_base::M_attached_to()`, and `__gnu_debug::Safe_iterator<_Iterator, ↵_Sequence>::__M_is_beginnest()`.

5.84.3.8 `template<typename _Iterator, typename _Sequence> bool __gnu_debug::Safe_iterator<_Iterator, _Sequence>::__M_dereferenceable () const [inline]`

Is the iterator dereferenceable?

Definition at line 427 of file `safe_iterator.h`.

Referenced by `__gnu_debug::__check_dereferenceable()`.

5.84.3.9 `void __gnu_debug::Safe_iterator_base::M_detach () [inherited]`

Detach the iterator for whatever sequence it is attached to, if any.

Referenced by `__gnu_debug::Safe_sequence_base::M_invalidate_all()`, and `__gnu_debug::Safe_iterator_base::↵_Safe_iterator_base()`.

5.84.3.10 `void __gnu_debug::Safe_iterator_base::M_detach_single () throw) [inherited]`

Likewise, but not thread-safe.

Referenced by `__gnu_debug::Safe_sequence_base::M_invalidate_all()`, `__gnu_debug::Safe_sequence< ↵_Sequence>::__M_transfer_from_if()`, and `__gnu_debug::Safe_iterator_base::_Safe_iterator_base()`.

5.84.3.11 `__gnu_cxx::__mutex& __gnu_debug::Safe_iterator_base::M_get_mutex () throw) [protected],`
`[inherited]`

For use in `_Safe_iterator`.

Referenced by `__gnu_debug::Safe_iterator_base::Safe_iterator_base()`, `__gnu_debug::Safe_local_iterator< ↵_Iterator, _Sequence>::__operator++()`, `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence>::__operator=()`, and `__gnu_debug::Safe_sequence_base::~~Safe_sequence_base()`.

5.84.3.12 `template<typename _Iterator, typename _Sequence> bool __gnu_debug::Safe_iterator<_Iterator, _Sequence>::__M_incrementable () const [inline]`

Is the iterator incrementable?

Definition at line 444 of file `safe_iterator.h`.

References `__gnu_debug::__check_dereferenceable()`.

5.84.3.13 `void __gnu_debug::_Safe_iterator_base::_M_invalidate () [inline],[inherited]`

Invalidate the iterator, making it singular.

Definition at line 144 of file `safe_base.h`.

References `__gnu_debug::_Safe_iterator_base::_M_reset()`.

5.84.3.14 `template<typename _Iterator, typename _Sequence> bool __gnu_debug::_Safe_iterator<_Iterator, _Sequence>::_M_is_before_begin () const [inline]`

Is this iterator equal to the sequence's `before_begin()` iterator if any?

Definition at line 483 of file `safe_iterator.h`.

Referenced by `__gnu_debug::__get_distance()`.

5.84.3.15 `template<typename _Iterator, typename _Sequence> bool __gnu_debug::_Safe_iterator<_Iterator, _Sequence>::_M_is_begin () const [inline]`

Is this iterator equal to the sequence's `begin()` iterator?

Definition at line 472 of file `safe_iterator.h`.

Referenced by `__gnu_debug::__get_distance()`.

5.84.3.16 `template<typename _Iterator, typename _Sequence> bool __gnu_debug::_Safe_iterator<_Iterator, _Sequence>::_M_is_beginnest () const [inline]`

Is this iterator equal to the sequence's `before_begin()` iterator if any or `begin()` otherwise?

Definition at line 489 of file `safe_iterator.h`.

References `__gnu_debug::_Safe_iterator_base::_M_can_compare()`, `__gnu_debug::_Safe_iterator_base::_M_is_singular()`, and `__gnu_debug::_Safe_iterator<_Iterator, _Sequence>::base()`.

5.84.3.17 `template<typename _Iterator, typename _Sequence> bool __gnu_debug::_Safe_iterator<_Iterator, _Sequence>::_M_is_end () const [inline]`

Is this iterator equal to the sequence's `end()` iterator?

Definition at line 477 of file `safe_iterator.h`.

Referenced by `__gnu_debug::__get_distance()`.

5.84.3.18 `void __gnu_debug::_Safe_iterator_base::_M_reset () throw) [inherited]`

Reset all member variables

Referenced by `__gnu_debug::_Safe_iterator_base::_M_invalidate()`.

5.84.3.19 `bool __gnu_debug::__Safe_iterator_base::_M_singular () const throw ()` `[inherited]`

Is this iterator singular?

Referenced by `__gnu_debug::__check_singular_aux()`, `__gnu_debug::__Safe_iterator_base::_M_attached_to()`, `__gnu_debug::__Safe_local_iterator<_Iterator, _Sequence>::_M_dereferenceable()`, `__gnu_debug::__Safe_local_iterator<_Iterator, _Sequence>::_M_incrementable()`, `__gnu_debug::__Safe_iterator<_Iterator, _Sequence>::_M_is_beginnest()`, and `__gnu_debug::__Safe_local_iterator<_Iterator, _Sequence>::_Safe_local_iterator()`.

5.84.3.20 `void __gnu_debug::__Safe_iterator_base::_M_unlink () throw ()` `[inline], [inherited]`

Unlink itself

Definition at line 153 of file `safe_base.h`.

References `__gnu_debug::__Safe_iterator_base::_M_next`, and `__gnu_debug::__Safe_iterator_base::_M_prior`.

5.84.3.21 `template<typename _Iterator, typename _Sequence> _Iterator& __gnu_debug::__Safe_iterator<_Iterator, _Sequence>::base ()` `[inline], [noexcept]`

Return the underlying iterator.

Definition at line 404 of file `safe_iterator.h`.

Referenced by `__gnu_debug::__get_distance()`, and `__gnu_debug::__Safe_iterator<_Iterator, _Sequence>::_M_is_beginnest()`.

5.84.3.22 `template<typename _Iterator, typename _Sequence> __gnu_debug::__Safe_iterator<_Iterator, _Sequence>::operator _Iterator () const` `[inline], [noexcept]`

Conversion to underlying non-debug iterator to allow better interaction with non-debug containers.

Definition at line 413 of file `safe_iterator.h`.

5.84.3.23 `template<typename _Iterator, typename _Sequence> reference __gnu_debug::__Safe_iterator<_Iterator, _Sequence>::operator* () const` `[inline], [noexcept]`

Iterator dereference.

Precondition

iterator is dereferenceable

Definition at line 266 of file `safe_iterator.h`.

5.84.3.24 `template<typename _Iterator, typename _Sequence> _Safe_iterator& __gnu_debug::_Safe_iterator< _Iterator, _Sequence >::operator++ () [inline], [noexcept]`

Iterator preincrement.

Precondition

iterator is incrementable

Definition at line 294 of file `safe_iterator.h`.

5.84.3.25 `template<typename _Iterator, typename _Sequence> _Safe_iterator __gnu_debug::_Safe_iterator< _Iterator, _Sequence >::operator++ (int) [inline], [noexcept]`

Iterator postincrement.

Precondition

iterator is incrementable

Definition at line 309 of file `safe_iterator.h`.

5.84.3.26 `template<typename _Iterator, typename _Sequence> _Safe_iterator& __gnu_debug::_Safe_iterator< _Iterator, _Sequence >::operator-- () [inline], [noexcept]`

Iterator predecrement.

Precondition

iterator is decrementable

Definition at line 324 of file `safe_iterator.h`.

5.84.3.27 `template<typename _Iterator, typename _Sequence> _Safe_iterator __gnu_debug::_Safe_iterator< _Iterator, _Sequence >::operator-- (int) [inline], [noexcept]`

Iterator postdecrement.

Precondition

iterator is decrementable

Definition at line 339 of file `safe_iterator.h`.

5.84.3.28 `template<typename _Iterator, typename _Sequence> pointer __gnu_debug::_Safe_iterator<_Iterator, _Sequence>::operator-> () const [inline], [noexcept]`

Iterator dereference.

Precondition

iterator is dereferenceable

Todo Make this correct w.r.t. iterators that return proxies

Definition at line 280 of file `safe_iterator.h`.

References `std::__addressof()`.

5.84.3.29 `template<typename _Iterator, typename _Sequence> _Safe_iterator& __gnu_debug::_Safe_iterator<_Iterator, _Sequence>::operator= (const _Safe_iterator<_Iterator, _Sequence> & __x) [inline], [noexcept]`

Copy assignment.

Definition at line 199 of file `safe_iterator.h`.

5.84.3.30 `template<typename _Iterator, typename _Sequence> _Safe_iterator& __gnu_debug::_Safe_iterator<_Iterator, _Sequence>::operator= (_Safe_iterator<_Iterator, _Sequence> && __x) [inline], [noexcept]`

Move assignment.

Postcondition

`__x` is singular and unattached

Definition at line 231 of file `safe_iterator.h`.

5.84.4 Member Data Documentation

5.84.4.1 `_Safe_iterator_base* __gnu_debug::_Safe_iterator_base::M_next [inherited]`

Pointer to the next iterator in the sequence's list of iterators. Only valid when `_M_sequence != NULL`.

Definition at line 72 of file `safe_base.h`.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::M_transfer_from_if()`, and `__gnu_debug::_Safe_iterator_base::M_unlink()`.

5.84.4.2 `_Safe_iterator_base*` `__gnu_debug::_Safe_iterator_base::_M_prior` `[inherited]`

Pointer to the previous iterator in the sequence's list of iterators. Only valid when `_M_sequence != NULL`.

Definition at line 68 of file `safe_base.h`.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`, and `__gnu_debug::_Safe_iterator_base::_M_unlink()`.

5.84.4.3 `_Safe_sequence_base*` `__gnu_debug::_Safe_iterator_base::_M_sequence` `[inherited]`

The sequence this iterator references; may be `NULL` to indicate a singular iterator.

Definition at line 55 of file `safe_base.h`.

Referenced by `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_incrementable()`, `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`, `__gnu_debug::_Safe_iterator_base::_Safe_iterator_base()`, `__gnu_debug::_Safe_local_iterator_base::_Safe_local_iterator_base()`, `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator++()`, and `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator=()`.

5.84.4.4 `unsigned int` `__gnu_debug::_Safe_iterator_base::_M_version` `[inherited]`

The version number of this iterator. The sentinel value 0 is used to indicate an invalidated iterator (i.e., one that is singular because of an operation on the container). This version number must equal the version number in the sequence referenced by `_M_sequence` for the iterator to be non-singular.

Definition at line 64 of file `safe_base.h`.

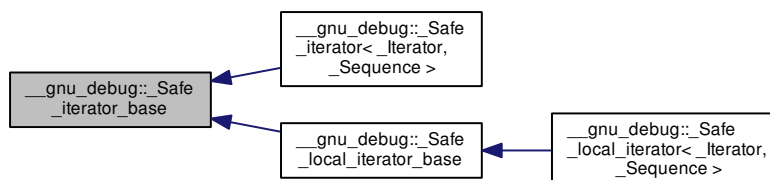
Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`, and `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator=()`.

The documentation for this class was generated from the following files:

- [formatter.h](#)
- [safe_iterator.h](#)
- [safe_iterator.tcc](#)

5.85 `__gnu_debug::_Safe_iterator_base` Class Reference

Inheritance diagram for `__gnu_debug::_Safe_iterator_base`:



Public Member Functions

- `void _M_attach (_Safe_sequence_base * __seq, bool __constant)`
- `void _M_attach_single (_Safe_sequence_base * __seq, bool __constant) throw ()`
- `bool _M_attached_to (const _Safe_sequence_base * __seq) const`
- `bool _M_can_compare (const _Safe_iterator_base & __x) const throw ()`
- `void _M_detach ()`
- `void _M_detach_single () throw ()`
- `void _M_invalidate ()`
- `void _M_reset () throw ()`
- `bool _M_singular () const throw ()`
- `void _M_unlink () throw ()`

Public Attributes

- `_Safe_iterator_base * _M_next`
- `_Safe_iterator_base * _M_prior`
- `_Safe_sequence_base * _M_sequence`
- `unsigned int _M_version`

Protected Member Functions

- `_Safe_iterator_base ()`
- `_Safe_iterator_base (const _Safe_sequence_base * __seq, bool __constant)`
- `_Safe_iterator_base (const _Safe_iterator_base & __x, bool __constant)`
- `__gnu_cxx::__mutex & _M_get_mutex () throw ()`

5.85.1 Detailed Description

Basic functionality for a *safe* iterator.

The `_Safe_iterator_base` base class implements the functionality of a safe iterator that is not specific to a particular iterator type. It contains a pointer back to the sequence it references along with iterator version information and pointers to form a doubly-linked list of iterators referenced by the container.

This class must not perform any operations that can throw an exception, or the exception guarantees of derived iterators will be broken.

Definition at line 50 of file `safe_base.h`.

5.85.2 Constructor & Destructor Documentation

5.85.2.1 `__gnu_debug::Safe_iterator_base::Safe_iterator_base ()` `[inline]`, `[protected]`

Initializes the iterator and makes it singular.

Definition at line 76 of file `safe_base.h`.

5.85.2.2 `__gnu_debug::Safe_iterator_base::Safe_iterator_base (const _Safe_sequence_base * __seq, bool __constant)`
`[inline], [protected]`

Initialize the iterator to reference the sequence pointed to by `__seq`. `__constant` is true when we are initializing a constant iterator, and false if it is a mutable iterator. Note that `__seq` may be NULL, in which case the iterator will be singular. Otherwise, the iterator will reference `__seq` and be nonsingular.

Definition at line 87 of file `safe_base.h`.

References `_M_attach()`.

5.85.2.3 `__gnu_debug::Safe_iterator_base::Safe_iterator_base (const _Safe_iterator_base & __x, bool __constant)`
`[inline], [protected]`

Initializes the iterator to reference the same sequence that `__x` does. `__constant` is true if this is a constant iterator, and false if it is mutable.

Definition at line 94 of file `safe_base.h`.

References `_M_attach()`, `_M_attach_single()`, `_M_detach()`, `_M_detach_single()`, `_M_get_mutex()`, and `_M_sequence`.

5.85.3 Member Function Documentation

5.85.3.1 `void __gnu_debug::Safe_iterator_base::M_attach (_Safe_sequence_base * __seq, bool __constant)`

Attaches this iterator to the given sequence, detaching it from whatever sequence it was attached to originally. If the new sequence is the NULL pointer, the iterator is left unattached.

Referenced by `__gnu_debug::Safe_sequence_base::M_invalidate_all()`, and `_Safe_iterator_base()`.

5.85.3.2 `void __gnu_debug::Safe_iterator_base::M_attach_single (_Safe_sequence_base * __seq, bool __constant) throw ()`

Likewise, but not thread-safe.

Referenced by `__gnu_debug::Safe_sequence_base::M_invalidate_all()`, and `_Safe_iterator_base()`.

5.85.3.3 `bool __gnu_debug::Safe_iterator_base::M_attached_to (const _Safe_sequence_base * __seq) const`
`[inline]`

Determines if we are attached to the given sequence.

Definition at line 129 of file `safe_base.h`.

References `_M_can_compare()`, and `_M_singular()`.

5.85.3.4 `bool __gnu_debug::Safe_iterator_base::M_can_compare (const _Safe_iterator_base & __x) const throw ()`

Can we compare this iterator to the given iterator `__x`? Returns true if both iterators are nonsingular and reference the same sequence.

Referenced by `_M_attached_to()`, and `__gnu_debug::Safe_iterator< _Iterator, _Sequence >::M_is_beginnest()`.

5.85.3.5 void __gnu_debug:: _Safe_iterator_base:: _M_detach ()

Detach the iterator for whatever sequence it is attached to, if any.

Referenced by __gnu_debug:: _Safe_sequence_base:: _M_invalidate_all(), and _Safe_iterator_base().

5.85.3.6 void __gnu_debug:: _Safe_iterator_base:: _M_detach_single () throw

Likewise, but not thread-safe.

Referenced by __gnu_debug:: _Safe_sequence_base:: _M_invalidate_all(), __gnu_debug:: _Safe_sequence< _Iterator, _Sequence >:: _M_transfer_from_if(), and _Safe_iterator_base().

5.85.3.7 __gnu_cxx:: _mutex& __gnu_debug:: _Safe_iterator_base:: _M_get_mutex () throw [protected]

For use in _Safe_iterator.

Referenced by _Safe_iterator_base(), __gnu_debug:: _Safe_local_iterator< _Iterator, _Sequence >:: operator++(), __gnu_debug:: _Safe_local_iterator< _Iterator, _Sequence >:: operator=(), and __gnu_debug:: _Safe_sequence_base::~ ~_Safe_sequence_base().

5.85.3.8 void __gnu_debug:: _Safe_iterator_base:: _M_invalidate () [inline]

Invalidate the iterator, making it singular.

Definition at line 144 of file safe_base.h.

References _M_reset().

5.85.3.9 void __gnu_debug:: _Safe_iterator_base:: _M_reset () throw

Reset all member variables

Referenced by _M_invalidate().

5.85.3.10 bool __gnu_debug:: _Safe_iterator_base:: _M_singular () const throw

Is this iterator singular?

Referenced by __gnu_debug:: _check_singular_aux(), _M_attached_to(), __gnu_debug:: _Safe_local_iterator< _Iterator, _Sequence >:: _M_dereferenceable(), __gnu_debug:: _Safe_local_iterator< _Iterator, _Sequence >:: _M_incrementable(), __gnu_debug:: _Safe_iterator< _Iterator, _Sequence >:: _M_is_beginnest(), and __gnu_debug:: _Safe_local_iterator< _Iterator, _Sequence >:: _Safe_local_iterator().

5.85.3.11 void __gnu_debug:: _Safe_iterator_base:: _M_unlink () throw [inline]

Unlink itself

Definition at line 153 of file safe_base.h.

References _M_next, and _M_prior.

5.85.4 Member Data Documentation

5.85.4.1 `_Safe_iterator_base* __gnu_debug::_Safe_iterator_base::_M_next`

Pointer to the next iterator in the sequence's list of iterators. Only valid when `_M_sequence != NULL`.

Definition at line 72 of file `safe_base.h`.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`, and `_M_unlink()`.

5.85.4.2 `_Safe_iterator_base* __gnu_debug::_Safe_iterator_base::_M_prior`

Pointer to the previous iterator in the sequence's list of iterators. Only valid when `_M_sequence != NULL`.

Definition at line 68 of file `safe_base.h`.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`, and `_M_unlink()`.

5.85.4.3 `_Safe_sequence_base* __gnu_debug::_Safe_iterator_base::_M_sequence`

The sequence this iterator references; may be `NULL` to indicate a singular iterator.

Definition at line 55 of file `safe_base.h`.

Referenced by `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_incrementable()`, `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`, `_Safe_iterator_base()`, `__gnu_debug::_Safe_local_iterator_base::_Safe_local_iterator_base()`, `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator++()`, and `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator=()`.

5.85.4.4 `unsigned int __gnu_debug::_Safe_iterator_base::_M_version`

The version number of this iterator. The sentinel value 0 is used to indicate an invalidated iterator (i.e., one that is singular because of an operation on the container). This version number must equal the version number in the sequence referenced by `_M_sequence` for the iterator to be non-singular.

Definition at line 64 of file `safe_base.h`.

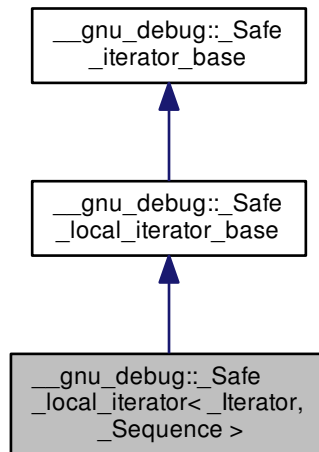
Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`, and `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator=()`.

The documentation for this class was generated from the following file:

- [safe_base.h](#)

5.86 `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>` Class Template Reference

Inheritance diagram for `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>`:



Public Types

- typedef `_Traits::difference_type` **difference_type**
- typedef `_Traits::iterator_category` **iterator_category**
- typedef `_Iterator` **iterator_type**
- typedef `_Traits::pointer` **pointer**
- typedef `_Traits::reference` **reference**
- typedef `_Traits::value_type` **value_type**

Public Member Functions

- `_Safe_local_iterator` () noexcept
- `_Safe_local_iterator` (const `_Iterator` &__i, const `_Safe_sequence_base` *__cont)
- `_Safe_local_iterator` (const `_Safe_local_iterator` &__x) noexcept
- `_Safe_local_iterator` (`_Safe_local_iterator` &&__x) noexcept
- template<typename `_MutableIterator` >
`_Safe_local_iterator` (const `_Safe_local_iterator`< `_MutableIterator`, typename `__gnu_cxx::__enable_if< std::__are_same< _MutableIterator, typename _Sequence::local_iterator::iterator_type >::__value, _Sequence >::__type >` &__x)
- void `_M_attach` (`_Safe_sequence_base` *__seq, bool __constant)
- void `_M_attach` (`_Safe_sequence_base` *__seq)
- void `_M_attach_single` (`_Safe_sequence_base` *__seq, bool __constant) throw ()
- void `_M_attach_single` (`_Safe_sequence_base` *__seq)

- `bool _M_attached_to (const _Safe_sequence_base * __seq) const`
- `bool _M_can_compare (const _Safe_iterator_base & __x) const throw ()`
- `bool _M_dereferenceable () const`
- `void _M_detach ()`
- `void _M_detach_single () throw ()`
- `__gnu_cxx::__conditional_type< std::__are_same< _Const_local_iterator, _Safe_local_iterator >::__value, const _Sequence *, _Sequence * >::__type _M_get_sequence () const`
- `template<typename _Other >`
`bool _M_in_same_bucket (const _Safe_local_iterator< _Other, _Sequence > & __other) const`
- `bool _M_incrementable () const`
- `void _M_invalidate ()`
- `bool _M_is_begin () const`
- `bool _M_is_end () const`
- `void _M_reset () throw ()`
- `bool _M_singular () const throw ()`
- `void _M_unlink () throw ()`
- `bool _M_valid_range (const _Safe_local_iterator & __rhs, std::pair< difference_type, _Distance_precision > & __dist_info) const`
- `_Iterator & base () noexcept`
- `const _Iterator & base () const noexcept`
- `size_type bucket () const`
- `operator _Iterator () const`
- `reference operator* () const`
- `_Safe_local_iterator & operator++ ()`
- `_Safe_local_iterator operator++ (int)`
- `pointer operator-> () const`
- `_Safe_local_iterator & operator= (const _Safe_local_iterator & __x)`
- `_Safe_local_iterator & operator= (_Safe_local_iterator && __x) noexcept`

Public Attributes

- `_Safe_iterator_base * _M_next`
- `_Safe_iterator_base * _M_prior`
- `_Safe_sequence_base * _M_sequence`
- `unsigned int _M_version`

Protected Member Functions

- `_Safe_unordered_container_base * _M_get_container () const noexcept`
- `__gnu_cxx::__mutex & _M_get_mutex () throw ()`

5.86.1 Detailed Description

```
template<typename _Iterator, typename _Sequence>
class __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >
```

Safe iterator wrapper.

The class template `_Safe_local_iterator` is a wrapper around an iterator that tracks the iterator's movement among sequences and checks that operations performed on the "safe" iterator are legal. In addition to the basic iterator operations (which are validated, and then passed to the underlying iterator), `_Safe_local_iterator` has member functions for iterator invalidation, attaching/detaching the iterator from sequences, and querying the iterator's state.

Definition at line 59 of file `formatter.h`.

5.86.2 Constructor & Destructor Documentation

5.86.2.1 `template<typename _Iterator, typename _Sequence> __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_Safe_local_iterator() [inline], [noexcept]`

Postcondition

the iterator is singular and unattached

Definition at line 84 of file `safe_local_iterator.h`.

Referenced by `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator++()`.

5.86.2.2 `template<typename _Iterator, typename _Sequence> __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_Safe_local_iterator(const _Iterator & __i, const _Safe_sequence_base* __cont) [inline]`

Safe iterator construction from an unsafe iterator and its sequence.

Precondition

`seq` is not NULL

Postcondition

this is not singular

Definition at line 93 of file `safe_local_iterator.h`.

References `__gnu_debug::_Safe_iterator_base::_M_singular()`.

5.86.2.3 `template<typename _Iterator, typename _Sequence> __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_Safe_local_iterator(const _Safe_local_iterator<_Iterator, _Sequence> & __x) [inline], [noexcept]`

Copy construction.

Definition at line 105 of file `safe_local_iterator.h`.

References `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_attach()`.

5.86.2.4 `template<typename _Iterator, typename _Sequence> __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_Safe_local_iterator(_Safe_local_iterator<_Iterator, _Sequence> && __x) [inline], [noexcept]`

Move construction.

Postcondition

`__x` is singular and unattached

Definition at line 122 of file `safe_local_iterator.h`.

References `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_attach()`, and `__gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::base()`.


```
5.86.2.5 template<typename _Iterator, typename _Sequence> template<typename _MutableIterator >
    __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_Safe_local_iterator ( const
        __gnu_debug::_Safe_local_iterator< _MutableIterator, typename __gnu_cxx::__enable_if< std::__are_same< _MutableIterator,
            typename _Sequence::local_iterator::iterator_type >::_value, _Sequence >::_type > & __x ) [inline]
```

Converting constructor from a mutable iterator to a constant iterator.

Definition at line 141 of file `safe_local_iterator.h`.

References `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_M_attach()`.

5.86.3 Member Function Documentation

```
5.86.3.1 void __gnu_debug::_Safe_local_iterator_base::_M_attach ( _Safe_sequence_base * __seq, bool __constant )
    [inherited]
```

Attaches this iterator to the given container, detaching it from whatever container it was attached to originally. If the new container is the NULL pointer, the iterator is left unattached.

Referenced by `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_M_attach()`, and `__gnu_debug::_Safe_local_iterator_base::_Safe_local_iterator_base()`.

```
5.86.3.2 template<typename _Iterator, typename _Sequence> void __gnu_debug::_Safe_local_iterator< _Iterator,
    _Sequence >::_M_attach ( _Safe_sequence_base * __seq ) [inline]
```

Attach iterator to the given sequence.

Definition at line 305 of file `safe_local_iterator.h`.

References `__gnu_debug::_Safe_local_iterator_base::_M_attach()`.

Referenced by `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_Safe_local_iterator()`, and `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator=()`.

```
5.86.3.3 void __gnu_debug::_Safe_local_iterator_base::_M_attach_single ( _Safe_sequence_base * __seq, bool __constant )
    throw ) [inherited]
```

Likewise, but not thread-safe.

Referenced by `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_M_attach_single()`, and `__gnu_debug::_Safe_local_iterator_base::_Safe_local_iterator_base()`.

```
5.86.3.4 template<typename _Iterator, typename _Sequence> void __gnu_debug::_Safe_local_iterator< _Iterator,
    _Sequence >::_M_attach_single ( _Safe_sequence_base * __seq ) [inline]
```

Likewise, but not thread-safe.

Definition at line 310 of file `safe_local_iterator.h`.

References `__gnu_debug::_Safe_local_iterator_base::_M_attach_single()`.

5.86.3.5 `bool __gnu_debug::Safe_iterator_base::M_attached_to (const _Safe_sequence_base * __seq) const`
`[inline], [inherited]`

Determines if we are attached to the given sequence.

Definition at line 129 of file `safe_base.h`.

References `__gnu_debug::Safe_iterator_base::M_can_compare()`, and `__gnu_debug::Safe_iterator_base::M_is_singular()`.

5.86.3.6 `bool __gnu_debug::Safe_iterator_base::M_can_compare (const _Safe_iterator_base & __x) const throw ()`
`[inherited]`

Can we compare this iterator to the given iterator `__x`? Returns true if both iterators are nonsingular and reference the same sequence.

Referenced by `__gnu_debug::Safe_iterator_base::M_attached_to()`, and `__gnu_debug::Safe_iterator<_Iterator, _Sequence>::M_is_beginnest()`.

5.86.3.7 `template<typename _Iterator, typename _Sequence> bool __gnu_debug::Safe_local_iterator<_Iterator, _Sequence>::M_dereferenceable () const` `[inline]`

Is the iterator dereferenceable?

Definition at line 315 of file `safe_local_iterator.h`.

References `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence>::M_is_end()`, and `__gnu_debug::Safe_iterator_base::M_singular()`.

Referenced by `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence>::operator*()`, and `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence>::operator->()`.

5.86.3.8 `void __gnu_debug::Safe_local_iterator_base::M_detach ()` `[inherited]`

Detach the iterator for whatever container it is attached to, if any.

Referenced by `__gnu_debug::Safe_local_iterator_base::Safe_local_iterator_base()`, and `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence>::operator=()`.

5.86.3.9 `void __gnu_debug::Safe_local_iterator_base::M_detach_single () throw ()` `[inherited]`

Likewise, but not thread-safe.

Referenced by `__gnu_debug::Safe_local_iterator_base::Safe_local_iterator_base()`.

5.86.3.10 `__gnu_cxx::mutex& __gnu_debug::Safe_iterator_base::M_get_mutex () throw ()` `[protected], [inherited]`

For use in `_Safe_iterator`.

Referenced by `__gnu_debug::Safe_iterator_base::Safe_iterator_base()`, `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence>::operator++()`, `__gnu_debug::Safe_local_iterator<_Iterator, _Sequence>::operator=()`, and `__gnu_debug::Safe_sequence_base::~~Safe_sequence_base()`.

5.86.3.11 `template<typename _Iterator, typename _Sequence> template<typename _Other > bool __gnu_debug::_Safe_↵
local_iterator< _Iterator, _Sequence >::_M_in_same_bucket (const _Safe_local_iterator< _Other, _Sequence >
& __other) const [inline]`

Is this iterator part of the same bucket as the other one?

Definition at line 349 of file `safe_local_iterator.h`.

References `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::bucket()`.

5.86.3.12 `template<typename _Iterator, typename _Sequence> bool __gnu_debug::_Safe_local_iterator< _Iterator,
_Sequence >::_M_incrementable () const [inline]`

Is the iterator incrementable?

Definition at line 320 of file `safe_local_iterator.h`.

References `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_M_is_end()`, `__gnu_debug::_Safe_↵
iterator_base::_M_sequence`, and `__gnu_debug::_Safe_iterator_base::_M_singular()`.

Referenced by `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator++()`.

5.86.3.13 `void __gnu_debug::_Safe_iterator_base::_M_invalidate () [inline], [inherited]`

Invalidate the iterator, making it singular.

Definition at line 144 of file `safe_base.h`.

References `__gnu_debug::_Safe_iterator_base::_M_reset()`.

5.86.3.14 `template<typename _Iterator, typename _Sequence> bool __gnu_debug::_Safe_local_iterator< _Iterator,
_Sequence >::_M_is_begin () const [inline]`

Is this iterator equal to the sequence's `begin(bucket)` iterator?

Definition at line 339 of file `safe_local_iterator.h`.

References `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::base()`, and `__gnu_debug::_Safe_local_↵
iterator< _Iterator, _Sequence >::bucket()`.

5.86.3.15 `template<typename _Iterator, typename _Sequence> bool __gnu_debug::_Safe_local_iterator< _Iterator,
_Sequence >::_M_is_end () const [inline]`

Is this iterator equal to the sequence's `end(bucket)` iterator?

Definition at line 343 of file `safe_local_iterator.h`.

References `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::base()`, and `__gnu_debug::_Safe_local_↵
iterator< _Iterator, _Sequence >::bucket()`.

Referenced by `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_M_dereferenceable()`, and `__gnu_↵
debug::_Safe_local_iterator< _Iterator, _Sequence >::_M_incrementable()`.

5.86.3.16 void __gnu_debug::_Safe_iterator_base::_M_reset () throw) [inherited]

Reset all member variables

Referenced by __gnu_debug::_Safe_iterator_base::_M_invalidate().

5.86.3.17 bool __gnu_debug::_Safe_iterator_base::_M_singular () const throw) [inherited]

Is this iterator singular?

Referenced by __gnu_debug::_check_singular_aux(), __gnu_debug::_Safe_iterator_base::_M_attached_to(), __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_dereferenceable(), __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_incrementable(), __gnu_debug::_Safe_iterator<_Iterator, _Sequence>::_M_is_beginnest(), and __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_Safe_local_iterator().

5.86.3.18 void __gnu_debug::_Safe_iterator_base::_M_unlink () throw) [inline],[inherited]

Unlink itself

Definition at line 153 of file safe_base.h.

References __gnu_debug::_Safe_iterator_base::_M_next, and __gnu_debug::_Safe_iterator_base::_M_prior.

5.86.3.19 template<typename _Iterator, typename _Sequence> _Iterator& __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::base () [inline],[noexcept]

Return the underlying iterator.

Definition at line 286 of file safe_local_iterator.h.

Referenced by __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_is_begin(), __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_is_end(), __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_Safe_local_iterator(), __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::bucket(), __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator*(), __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator++(), __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator->(), and __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::operator=().

5.86.3.20 template<typename _Iterator, typename _Sequence> size_type __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::bucket () const [inline]

Return the bucket.

Definition at line 295 of file safe_local_iterator.h.

References __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::base().

Referenced by __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_in_same_bucket(), __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_is_begin(), and __gnu_debug::_Safe_local_iterator<_Iterator, _Sequence>::_M_is_end().

5.86.3.21 `template<typename _Iterator, typename _Sequence> __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator _Iterator () const [inline]`

Conversion to underlying non-debug iterator to allow better interaction with non-debug containers.

Definition at line 301 of file `safe_local_iterator.h`.

5.86.3.22 `template<typename _Iterator, typename _Sequence> reference __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator*() const [inline]`

Iterator dereference.

Precondition

iterator is dereferenceable

Definition at line 228 of file `safe_local_iterator.h`.

References `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::M_dereferenceable()`, and `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::base()`.

5.86.3.23 `template<typename _Iterator, typename _Sequence> _Safe_local_iterator& __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator++ () [inline]`

Iterator preincrement.

Precondition

iterator is incrementable

Definition at line 256 of file `safe_local_iterator.h`.

References `__gnu_debug::_Safe_iterator_base::M_get_mutex()`, `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::M_incrementable()`, and `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::base()`.

5.86.3.24 `template<typename _Iterator, typename _Sequence> _Safe_local_iterator __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator++ (int) [inline]`

Iterator postincrement.

Precondition

iterator is incrementable

Definition at line 271 of file `safe_local_iterator.h`.

References `__gnu_debug::_Safe_iterator_base::M_get_mutex()`, `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::M_incrementable()`, `__gnu_debug::_Safe_iterator_base::M_sequence`, `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::Safe_local_iterator()`, and `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::base()`.

5.86.3.25 `template<typename _Iterator, typename _Sequence> pointer __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator-> () const [inline]`

Iterator dereference.

Precondition

iterator is dereferenceable

Todo Make this correct w.r.t. iterators that return proxies

Definition at line 242 of file `safe_local_iterator.h`.

References `std::__addressof()`, `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::M_dereferenceable()`, and `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::base()`.

5.86.3.26 `template<typename _Iterator, typename _Sequence> _Safe_local_iterator& __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator= (const _Safe_local_iterator< _Iterator, _Sequence > & __x) [inline]`

Copy assignment.

Definition at line 163 of file `safe_local_iterator.h`.

References `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::M_attach()`, `__gnu_debug::_Safe_local_iterator_base::M_detach()`, `__gnu_debug::_Safe_iterator_base::M_get_mutex()`, `__gnu_debug::_Safe_iterator_base::M_sequence`, `__gnu_debug::_Safe_iterator_base::M_version`, and `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::base()`.

5.86.3.27 `template<typename _Iterator, typename _Sequence> _Safe_local_iterator& __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator= (_Safe_local_iterator< _Iterator, _Sequence > && __x) [inline], [noexcept]`

Move assignment.

Postcondition

`__x` is singular and unattached

Definition at line 194 of file `safe_local_iterator.h`.

References `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::M_attach()`, `__gnu_debug::_Safe_local_iterator_base::M_detach()`, `__gnu_debug::_Safe_iterator_base::M_get_mutex()`, `__gnu_debug::_Safe_iterator_base::M_sequence`, `__gnu_debug::_Safe_iterator_base::M_version`, and `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::base()`.

5.86.4 Member Data Documentation

5.86.4.1 `_Safe_iterator_base* __gnu_debug::_Safe_iterator_base::M_next` `[inherited]`

Pointer to the next iterator in the sequence's list of iterators. Only valid when `_M_sequence != NULL`.

Definition at line 72 of file `safe_base.h`.

Referenced by `__gnu_debug::_Safe_sequence< _Sequence >::M_transfer_from_if()`, and `__gnu_debug::_Safe_iterator_base::M_unlink()`.

5.86.4.2 `_Safe_iterator_base* __gnu_debug::_Safe_iterator_base::M_prior` `[inherited]`

Pointer to the previous iterator in the sequence's list of iterators. Only valid when `_M_sequence != NULL`.

Definition at line 68 of file `safe_base.h`.

Referenced by `__gnu_debug::_Safe_sequence< _Sequence >::M_transfer_from_if()`, and `__gnu_debug::_Safe_iterator_base::M_unlink()`.

5.86.4.3 `_Safe_sequence_base* __gnu_debug::_Safe_iterator_base::M_sequence` `[inherited]`

The sequence this iterator references; may be `NULL` to indicate a singular iterator.

Definition at line 55 of file `safe_base.h`.

Referenced by `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::M_incrementable()`, `__gnu_debug::_Safe_sequence< _Sequence >::M_transfer_from_if()`, `__gnu_debug::_Safe_iterator_base::Safe_iterator_base()`, `__gnu_debug::_Safe_local_iterator_base::Safe_local_iterator_base()`, `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator++()`, and `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator=()`.

5.86.4.4 `unsigned int __gnu_debug::_Safe_iterator_base::M_version` `[inherited]`

The version number of this iterator. The sentinel value 0 is used to indicate an invalidated iterator (i.e., one that is singular because of an operation on the container). This version number must equal the version number in the sequence referenced by `_M_sequence` for the iterator to be non-singular.

Definition at line 64 of file `safe_base.h`.

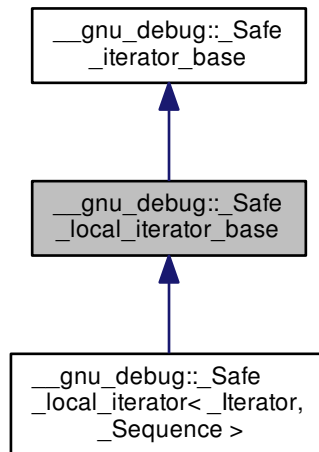
Referenced by `__gnu_debug::_Safe_sequence< _Sequence >::M_transfer_from_if()`, and `__gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator=()`.

The documentation for this class was generated from the following files:

- [formatter.h](#)
- [safe_local_iterator.h](#)
- [safe_local_iterator.tcc](#)

5.87 __gnu_debug::_Safe_local_iterator_base Class Reference

Inheritance diagram for __gnu_debug::_Safe_local_iterator_base:



Public Member Functions

- void [_M_attach](#) ([_Safe_sequence_base](#) * __seq, bool __constant)
- void [_M_attach_single](#) ([_Safe_sequence_base](#) * __seq, bool __constant) throw ()
- bool [_M_attached_to](#) (const [_Safe_sequence_base](#) * __seq) const
- bool [_M_can_compare](#) (const [_Safe_iterator_base](#) & __x) const throw ()
- void [_M_detach](#) ()
- void [_M_detach_single](#) () throw ()
- void [_M_invalidate](#) ()
- void [_M_reset](#) () throw ()
- bool [_M_singular](#) () const throw ()
- void [_M_unlink](#) () throw ()

Public Attributes

- [_Safe_iterator_base](#) * [_M_next](#)
- [_Safe_iterator_base](#) * [_M_prior](#)
- [_Safe_sequence_base](#) * [_M_sequence](#)
- unsigned int [_M_version](#)

Protected Member Functions

- `_Safe_local_iterator_base()`
- `_Safe_local_iterator_base(const _Safe_sequence_base * __seq, bool __constant)`
- `_Safe_local_iterator_base(const _Safe_local_iterator_base & __x, bool __constant)`
- `_Safe_unordered_container_base * _M_get_container() const noexcept`
- `__gnu_cxx::__mutex & _M_get_mutex() throw()`

5.87.1 Detailed Description

Basic functionality for a *safe* iterator.

The `_Safe_local_iterator_base` base class implements the functionality of a safe local iterator that is not specific to a particular iterator type. It contains a pointer back to the container it references along with iterator version information and pointers to form a doubly-linked list of local iterators referenced by the container.

This class must not perform any operations that can throw an exception, or the exception guarantees of derived iterators will be broken.

Definition at line 50 of file `safe_unordered_base.h`.

5.87.2 Constructor & Destructor Documentation

5.87.2.1 `__gnu_debug::_Safe_local_iterator_base::_Safe_local_iterator_base()` `[inline]`, `[protected]`

Initializes the iterator and makes it singular.

Definition at line 54 of file `safe_unordered_base.h`.

5.87.2.2 `__gnu_debug::_Safe_local_iterator_base::_Safe_local_iterator_base(const _Safe_sequence_base * __seq, bool __constant)` `[inline]`, `[protected]`

Initialize the iterator to reference the container pointed to by `__seq`. `__constant` is true when we are initializing a constant local iterator, and false if it is a mutable local iterator. Note that `__seq` may be NULL, in which case the iterator will be singular. Otherwise, the iterator will reference `__seq` and be nonsingular.

Definition at line 64 of file `safe_unordered_base.h`.

References `_M_attach()`.

5.87.2.3 `__gnu_debug::_Safe_local_iterator_base::_Safe_local_iterator_base(const _Safe_local_iterator_base & __x, bool __constant)` `[inline]`, `[protected]`

Initializes the iterator to reference the same container that `__x` does. `__constant` is true if this is a constant iterator, and false if it is mutable.

Definition at line 70 of file `safe_unordered_base.h`.

References `_M_attach()`, `_M_attach_single()`, `_M_detach()`, `_M_detach_single()`, and `__gnu_debug::_Safe_iterator_base::_M_sequence`.

5.87.3 Member Function Documentation

5.87.3.1 void __gnu_debug::_Safe_local_iterator_base::_M_attach (_Safe_sequence_base * __seq, bool __constant)

Attaches this iterator to the given container, detaching it from whatever container it was attached to originally. If the new container is the NULL pointer, the iterator is left unattached.

Referenced by __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_M_attach(), and _Safe_local_iterator_↔base().

5.87.3.2 void __gnu_debug::_Safe_local_iterator_base::_M_attach_single (_Safe_sequence_base * __seq, bool __constant) throw)

Likewise, but not thread-safe.

Referenced by __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_M_attach_single(), and _Safe_local_↔iterator_base().

5.87.3.3 bool __gnu_debug::_Safe_iterator_base::_M_attached_to (const _Safe_sequence_base * __seq) const
[inline], [inherited]

Determines if we are attached to the given sequence.

Definition at line 129 of file safe_base.h.

References __gnu_debug::_Safe_iterator_base::_M_can_compare(), and __gnu_debug::_Safe_iterator_base::_M_↔singular().

5.87.3.4 bool __gnu_debug::_Safe_iterator_base::_M_can_compare (const _Safe_iterator_base & __x) const throw)
[inherited]

Can we compare this iterator to the given iterator __x? Returns true if both iterators are nonsingular and reference the same sequence.

Referenced by __gnu_debug::_Safe_iterator_base::_M_attached_to(), and __gnu_debug::_Safe_iterator< _Iterator, _↔Sequence >::_M_is_beginnest().

5.87.3.5 void __gnu_debug::_Safe_local_iterator_base::_M_detach ()

Detach the iterator for whatever container it is attached to, if any.

Referenced by _Safe_local_iterator_base(), and __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >_↔::operator=().

5.87.3.6 void __gnu_debug::_Safe_local_iterator_base::_M_detach_single () throw)

Likewise, but not thread-safe.

Referenced by _Safe_local_iterator_base().

5.87.3.7 `__gnu_cxx::mutex& __gnu_debug::Safe_iterator_base::M_get_mutex () throw` `[protected]`,
`[inherited]`

For use in `_Safe_iterator`.

Referenced by `__gnu_debug::Safe_iterator_base::Safe_iterator_base()`, `__gnu_debug::Safe_local_iterator< _Iterator, _Sequence >::operator++()`, `__gnu_debug::Safe_local_iterator< _Iterator, _Sequence >::operator=()`, and `__gnu_debug::Safe_sequence_base::~~Safe_sequence_base()`.

5.87.3.8 `void __gnu_debug::Safe_iterator_base::M_invalidate ()` `[inline]`, `[inherited]`

Invalidate the iterator, making it singular.

Definition at line 144 of file `safe_base.h`.

References `__gnu_debug::Safe_iterator_base::M_reset()`.

5.87.3.9 `void __gnu_debug::Safe_iterator_base::M_reset () throw` `[inherited]`

Reset all member variables

Referenced by `__gnu_debug::Safe_iterator_base::M_invalidate()`.

5.87.3.10 `bool __gnu_debug::Safe_iterator_base::M_singular () const throw` `[inherited]`

Is this iterator singular?

Referenced by `__gnu_debug::check_singular_aux()`, `__gnu_debug::Safe_iterator_base::M_attached_to()`, `__gnu_debug::Safe_local_iterator< _Iterator, _Sequence >::M_dereferenceable()`, `__gnu_debug::Safe_local_iterator< _Iterator, _Sequence >::M_incrementable()`, `__gnu_debug::Safe_iterator< _Iterator, _Sequence >::M_is_beginnest()`, and `__gnu_debug::Safe_local_iterator< _Iterator, _Sequence >::Safe_local_iterator()`.

5.87.3.11 `void __gnu_debug::Safe_iterator_base::M_unlink () throw` `[inline]`, `[inherited]`

Unlink itself

Definition at line 153 of file `safe_base.h`.

References `__gnu_debug::Safe_iterator_base::M_next`, and `__gnu_debug::Safe_iterator_base::M_prior`.

5.87.4 Member Data Documentation

5.87.4.1 `_Safe_iterator_base* __gnu_debug::Safe_iterator_base::M_next` `[inherited]`

Pointer to the next iterator in the sequence's list of iterators. Only valid when `_M_sequence != NULL`.

Definition at line 72 of file `safe_base.h`.

Referenced by `__gnu_debug::Safe_sequence< _Sequence >::M_transfer_from_if()`, and `__gnu_debug::Safe_iterator_base::M_unlink()`.

5.87.4.2 _Safe_iterator_base* __gnu_debug::_Safe_iterator_base::_M_prior [inherited]

Pointer to the previous iterator in the sequence's list of iterators. Only valid when _M_sequence != NULL.

Definition at line 68 of file safe_base.h.

Referenced by __gnu_debug::_Safe_sequence< _Sequence >::_M_transfer_from_if(), and __gnu_debug::_Safe_iterator_base::_M_unlink().

5.87.4.3 _Safe_sequence_base* __gnu_debug::_Safe_iterator_base::_M_sequence [inherited]

The sequence this iterator references; may be NULL to indicate a singular iterator.

Definition at line 55 of file safe_base.h.

Referenced by __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::_M_incrementable(), __gnu_debug::_Safe_sequence< _Sequence >::_M_transfer_from_if(), __gnu_debug::_Safe_iterator_base::_Safe_iterator_base(), __gnu_debug::_Safe_local_iterator_base(), __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator++(), and __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator=().

5.87.4.4 unsigned int __gnu_debug::_Safe_iterator_base::_M_version [inherited]

The version number of this iterator. The sentinel value 0 is used to indicate an invalidated iterator (i.e., one that is singular because of an operation on the container). This version number must equal the version number in the sequence referenced by _M_sequence for the iterator to be non-singular.

Definition at line 64 of file safe_base.h.

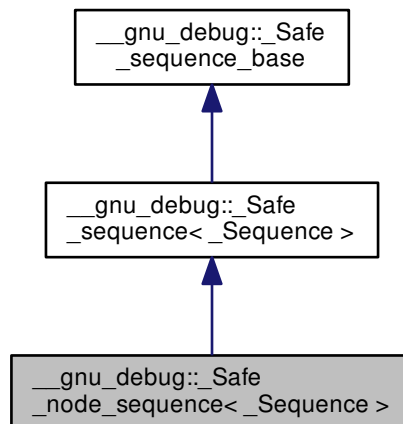
Referenced by __gnu_debug::_Safe_sequence< _Sequence >::_M_transfer_from_if(), and __gnu_debug::_Safe_local_iterator< _Iterator, _Sequence >::operator=().

The documentation for this class was generated from the following file:

- [safe_unordered_base.h](#)

5.88 __gnu_debug::_Safe_node_sequence< _Sequence > Class Template Reference

Inheritance diagram for __gnu_debug::_Safe_node_sequence< _Sequence >:



Public Member Functions

- void [_M_attach](#) ([_Safe_iterator_base](#) * __it, bool __constant)
- void [_M_attach_single](#) ([_Safe_iterator_base](#) * __it, bool __constant) throw ()
- void [_M_detach](#) ([_Safe_iterator_base](#) * __it)
- void [_M_detach_single](#) ([_Safe_iterator_base](#) * __it) throw ()
- void [_M_invalidate_all](#) () const
- template<typename [_Predicate](#) >
void [_M_invalidate_if](#) ([_Predicate](#) __pred)
- template<typename [_Predicate](#) >
void [_M_transfer_from_if](#) ([_Safe_sequence](#) & __from, [_Predicate](#) __pred)

Public Attributes

- [_Safe_iterator_base](#) * [_M_const_iterators](#)
- [_Safe_iterator_base](#) * [_M_iterators](#)
- unsigned int [_M_version](#)

Protected Member Functions

- void [_M_detach_all](#) ()
- void [_M_detach_singular](#) ()
- [__gnu_cxx::__mutex](#) & [_M_get_mutex](#) () throw ()
- void [_M_invalidate_all](#) ()
- void [_M_revalidate_singular](#) ()
- void [_M_swap](#) ([_Safe_sequence_base](#) & __x) noexcept

5.88.1 Detailed Description

```
template<typename \_Sequence>
class \_\_gnu\_debug::\_Safe\_node\_sequence< \_Sequence >
```

Like [_Safe_sequence](#) but with a special [_M_invalidate_all](#) implementation not invalidating past-the-end iterators. Used by node based sequence.

Definition at line 131 of file [safe_sequence.h](#).

5.88.2 Member Function Documentation

5.88.2.1 void [__gnu_debug::_Safe_sequence_base::_M_attach](#) ([_Safe_iterator_base](#) * __it, bool __constant)
[inherited]

Attach an iterator to this sequence.

5.88.2.2 void __gnu_debug::_Safe_sequence_base::M_attach_single (_Safe_iterator_base * __it, bool __constant) throw)
[inherited]

Likewise but not thread safe.

5.88.2.3 void __gnu_debug::_Safe_sequence_base::M_detach (_Safe_iterator_base * __it) [inherited]

Detach an iterator from this sequence

Referenced by __gnu_debug::_Safe_iterator< _Iterator, _Sequence >::_Safe_iterator().

5.88.2.4 void __gnu_debug::_Safe_sequence_base::M_detach_all () [protected],[inherited]

Detach all iterators, leaving them singular.

5.88.2.5 void __gnu_debug::_Safe_sequence_base::M_detach_single (_Safe_iterator_base * __it) throw)
[inherited]

Likewise but not thread safe.

5.88.2.6 void __gnu_debug::_Safe_sequence_base::M_detach_singular () [protected],[inherited]

Detach all singular iterators.

Postcondition

for all iterators i attached to this sequence, i->_M_version == _M_version.

5.88.2.7 __gnu_cxx::mutex& __gnu_debug::_Safe_sequence_base::M_get_mutex () throw) [protected],
[inherited]

For use in _Safe_sequence.

Referenced by __gnu_debug::_Safe_sequence< _Sequence >::M_transfer_from_if().

5.88.2.8 void __gnu_debug::_Safe_sequence_base::M_invalidate_all () const [inline],[inherited]

Invalidates all iterators.

Definition at line 248 of file safe_base.h.

References __gnu_debug::_Safe_iterator_base::M_attach(), __gnu_debug::_Safe_iterator_base::M_attach_single(), __gnu_debug::_Safe_iterator_base::M_detach(), and __gnu_debug::_Safe_iterator_base::M_detach_single().

5.88.2.9 `template<typename _Sequence > template<typename _Predicate > void __gnu_debug::_Safe_sequence<_Sequence >::_M_invalidate_if (_Predicate __pred) [inherited]`

Invalidates all iterators `x` that reference this sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

Definition at line 38 of file `safe_sequence.tcc`.

References `__gnu_debug::_Safe_sequence<_Sequence >::_M_transfer_from_if()`.

5.88.2.10 `void __gnu_debug::_Safe_sequence_base::_M_revalidate_singular () [protected],[inherited]`

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

5.88.2.11 `void __gnu_debug::_Safe_sequence_base::_M_swap (_Safe_sequence_base & __x) [protected],[noexcept],[inherited]`

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.88.2.12 `template<typename _Sequence > template<typename _Predicate > void __gnu_debug::_Safe_sequence<_Sequence >::_M_transfer_from_if (_Safe_sequence<_Sequence > & __from, _Predicate __pred) [inherited]`

Transfers all iterators `x` that reference `from` sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

Definition at line 69 of file `safe_sequence.tcc`.

References `__gnu_debug::_Safe_sequence_base::_M_const_iterators`, `__gnu_debug::_Safe_iterator_base::_M_detach_single()`, `__gnu_debug::_Safe_sequence_base::_M_get_mutex()`, `__gnu_debug::_Safe_sequence_base::_M_iterators`, `__gnu_debug::_Safe_iterator_base::_M_next`, `__gnu_debug::_Safe_iterator_base::_M_prior`, `__gnu_debug::_Safe_iterator_base::_M_sequence`, and `__gnu_debug::_Safe_iterator_base::_M_version`.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence >::_M_invalidate_if()`.

5.88.3 Member Data Documentation

5.88.3.1 `_Safe_iterator_base* __gnu_debug::_Safe_sequence_base::_M_const_iterators [inherited]`

The list of constant iterators that reference this container.

Definition at line 193 of file `safe_base.h`.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence >::_M_transfer_from_if()`.

5.88.3.2 __Safe_iterator_base* __gnu_debug::__Safe_sequence_base::__M_iterators [inherited]

The list of mutable iterators that reference this container.

Definition at line 190 of file safe_base.h.

Referenced by __gnu_debug::__Safe_sequence<_Sequence>::__M_transfer_from_if().

5.88.3.3 unsigned int __gnu_debug::__Safe_sequence_base::__M_version [mutable],[inherited]

The container version number. This number may never be 0.

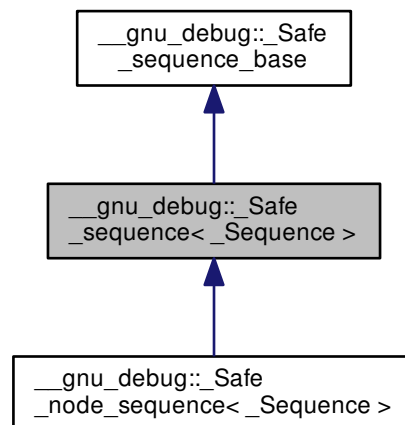
Definition at line 196 of file safe_base.h.

The documentation for this class was generated from the following file:

- [safe_sequence.h](#)

5.89 __gnu_debug::__Safe_sequence<_Sequence> Class Template Reference

Inheritance diagram for __gnu_debug::__Safe_sequence<_Sequence>:



Public Member Functions

- void [_M_attach](#) ([_Safe_iterator_base](#) * __it, bool __constant)
- void [_M_attach_single](#) ([_Safe_iterator_base](#) * __it, bool __constant) throw ()
- void [_M_detach](#) ([_Safe_iterator_base](#) * __it)
- void [_M_detach_single](#) ([_Safe_iterator_base](#) * __it) throw ()
- void [_M_invalidate_all](#) () const
- template<typename _Predicate>
void [_M_invalidate_if](#) (_Predicate __pred)
- template<typename _Predicate>
void [_M_transfer_from_if](#) ([_Safe_sequence](#) & __from, _Predicate __pred)

Public Attributes

- [_Safe_iterator_base](#) * [_M_const_iterators](#)
- [_Safe_iterator_base](#) * [_M_iterators](#)
- unsigned int [_M_version](#)

Protected Member Functions

- void [_M_detach_all](#) ()
- void [_M_detach_singular](#) ()
- [__gnu_cxx::__mutex](#) & [_M_get_mutex](#) () throw ()
- void [_M_revalidate_singular](#) ()
- void [_M_swap](#) ([_Safe_sequence_base](#) &__x) noexcept

5.89.1 Detailed Description

```
template<typename _Sequence>
class __gnu_debug::_Safe_sequence< _Sequence >
```

Base class for constructing a *safe* sequence type that tracks iterators that reference it.

The class template `_Safe_sequence` simplifies the construction of *safe* sequences that track the iterators that reference the sequence, so that the iterators are notified of changes in the sequence that may affect their operation, e.g., if the container invalidates its iterators or is destructed. This class template may only be used by deriving from it and passing the name of the derived class as its template parameter via the curiously recurring template pattern. The derived class must have `iterator` and `const_iterator` types that are instantiations of class template `_Safe_iterator` for this sequence. Iterators will then be tracked automatically.

Definition at line 62 of file `formatter.h`.

5.89.2 Member Function Documentation

5.89.2.1 void `__gnu_debug::_Safe_sequence_base::_M_attach (_Safe_iterator_base * __it, bool __constant)`
[inherited]

Attach an iterator to this sequence.

5.89.2.2 void `__gnu_debug::_Safe_sequence_base::_M_attach_single (_Safe_iterator_base * __it, bool __constant) throw`
[inherited]

Likewise but not thread safe.

5.89.2.3 void `__gnu_debug::_Safe_sequence_base::_M_detach (_Safe_iterator_base * __it)` [inherited]

Detach an iterator from this sequence

Referenced by `__gnu_debug::_Safe_iterator< _Iterator, _Sequence >::_Safe_iterator()`.

5.89.2.4 void __gnu_debug::_Safe_sequence_base::_M_detach_all() [protected],[inherited]

Detach all iterators, leaving them singular.

5.89.2.5 void __gnu_debug::_Safe_sequence_base::_M_detach_single(_Safe_iterator_base * __it) throw
[inherited]

Likewise but not thread safe.

5.89.2.6 void __gnu_debug::_Safe_sequence_base::_M_detach_singular() [protected],[inherited]

Detach all singular iterators.

Postcondition

for all iterators i attached to this sequence, i->_M_version == _M_version.

5.89.2.7 __gnu_cxx::mutex& __gnu_debug::_Safe_sequence_base::_M_get_mutex() throw [protected],
[inherited]

For use in _Safe_sequence.

Referenced by __gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if().

5.89.2.8 void __gnu_debug::_Safe_sequence_base::_M_invalidate_all() const [inline],[inherited]

Invalidates all iterators.

Definition at line 248 of file safe_base.h.

References __gnu_debug::_Safe_iterator_base::_M_attach(), __gnu_debug::_Safe_iterator_base::_M_attach_single(),
__gnu_debug::_Safe_iterator_base::_M_detach(), and __gnu_debug::_Safe_iterator_base::_M_detach_single().

5.89.2.9 template<typename _Sequence> template<typename _Predicate> void __gnu_debug::_Safe_sequence<
_Sequence>::_M_invalidate_if(_Predicate __pred)

Invalidates all iterators x that reference this sequence, are not singular, and for which __pred(x) returns true.
__pred will be invoked with the normal iterators nested in the safe ones.

Definition at line 38 of file safe_sequence.tcc.

References __gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if().

5.89.2.10 void __gnu_debug::_Safe_sequence_base::_M_revalidate_singular() [protected],[inherited]

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before
(but for some reason, such as an exception, need to become valid again).

5.89.2.11 `void __gnu_debug::Safe_sequence_base::_M_swap (_Safe_sequence_base & __x) [protected], [noexcept], [inherited]`

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.89.2.12 `template<typename _Sequence > template<typename _Predicate > void __gnu_debug::Safe_sequence<_Sequence>::_M_transfer_from_if (_Safe_sequence<_Sequence> & __from, _Predicate __pred)`

Transfers all iterators `x` that reference `from` sequence, are not singular, and for which `__pred(x)` returns `true`. `__pred` will be invoked with the normal iterators nested in the safe ones.

Definition at line 69 of file `safe_sequence.tcc`.

References `__gnu_debug::Safe_sequence_base::_M_const_iterators`, `__gnu_debug::Safe_iterator_base::_M_detach_single()`, `__gnu_debug::Safe_sequence_base::_M_get_mutex()`, `__gnu_debug::Safe_sequence_base::_M_iterators`, `__gnu_debug::Safe_iterator_base::_M_next`, `__gnu_debug::Safe_iterator_base::_M_prior`, `__gnu_debug::Safe_iterator_base::_M_sequence`, and `__gnu_debug::Safe_iterator_base::_M_version`.

Referenced by `__gnu_debug::Safe_sequence<_Sequence>::_M_invalidate_if()`.

5.89.3 Member Data Documentation

5.89.3.1 `_Safe_iterator_base* __gnu_debug::Safe_sequence_base::_M_const_iterators [inherited]`

The list of constant iterators that reference this container.

Definition at line 193 of file `safe_base.h`.

Referenced by `__gnu_debug::Safe_sequence<_Sequence>::_M_transfer_from_if()`.

5.89.3.2 `_Safe_iterator_base* __gnu_debug::Safe_sequence_base::_M_iterators [inherited]`

The list of mutable iterators that reference this container.

Definition at line 190 of file `safe_base.h`.

Referenced by `__gnu_debug::Safe_sequence<_Sequence>::_M_transfer_from_if()`.

5.89.3.3 `unsigned int __gnu_debug::Safe_sequence_base::_M_version [mutable], [inherited]`

The container version number. This number may never be 0.

Definition at line 196 of file `safe_base.h`.

The documentation for this class was generated from the following files:

- [formatter.h](#)
- [safe_sequence.h](#)
- [safe_sequence.tcc](#)

Protected Member Functions

- `_Safe_sequence_base` (const `_Safe_sequence_base` &) noexcept
- `~_Safe_sequence_base` ()
- `void _M_detach_all` ()
- `void _M_detach_singular` ()
- `__gnu_cxx::__mutex & _M_get_mutex` () throw ()
- `void _M_revalidate_singular` ()
- `void _M_swap` (`_Safe_sequence_base` &__x) noexcept

5.90.1 Detailed Description

Base class that supports tracking of iterators that reference a sequence.

The `_Safe_sequence_base` class provides basic support for tracking iterators into a sequence. Sequences that track iterators must derived from `_Safe_sequence_base` publicly, so that safe iterators (which inherit `_Safe_iterator_base`) can attach to them. This class contains two linked lists of iterators, one for constant iterators and one for mutable iterators, and a version number that allows very fast invalidation of all iterators that reference the container.

This class must ensure that no operation on it may throw an exception, otherwise *safe* sequences may fail to provide the exception-safety guarantees required by the C++ standard.

Definition at line 186 of file `safe_base.h`.

5.90.2 Constructor & Destructor Documentation

5.90.2.1 `__gnu_debug::_Safe_sequence_base::~~_Safe_sequence_base` () `[inline]`, `[protected]`

Notify all iterators that reference this sequence that the sequence is being destroyed.

Definition at line 211 of file `safe_base.h`.

References `__gnu_debug::_Safe_iterator_base::_M_get_mutex`().

5.90.3 Member Function Documentation

5.90.3.1 `void __gnu_debug::_Safe_sequence_base::_M_attach` (`_Safe_iterator_base` * `__it`, bool `__constant`)

Attach an iterator to this sequence.

5.90.3.2 `void __gnu_debug::_Safe_sequence_base::_M_attach_single` (`_Safe_iterator_base` * `__it`, bool `__constant`) throw)

Likewise but not thread safe.

5.90.3.3 `void __gnu_debug::Safe_sequence_base::M_detach (_Safe_iterator_base * __it)`

Detach an iterator from this sequence

Referenced by `__gnu_debug::Safe_iterator< _Iterator, _Sequence >::Safe_iterator()`.

5.90.3.4 `void __gnu_debug::Safe_sequence_base::M_detach_all ()` `[protected]`

Detach all iterators, leaving them singular.

5.90.3.5 `void __gnu_debug::Safe_sequence_base::M_detach_single (_Safe_iterator_base * __it) throw`

Likewise but not thread safe.

5.90.3.6 `void __gnu_debug::Safe_sequence_base::M_detach_singular ()` `[protected]`

Detach all singular iterators.

Postcondition

for all iterators `i` attached to this sequence, `i->_M_version == _M_version`.

5.90.3.7 `__gnu_cxx::mutex& __gnu_debug::Safe_sequence_base::M_get_mutex () throw` `[protected]`

For use in `_Safe_sequence`.

Referenced by `__gnu_debug::Safe_sequence< _Sequence >::M_transfer_from_if()`.

5.90.3.8 `void __gnu_debug::Safe_sequence_base::M_invalidate_all () const` `[inline]`

Invalidates all iterators.

Definition at line 248 of file `safe_base.h`.

References `__gnu_debug::Safe_iterator_base::M_attach()`, `__gnu_debug::Safe_iterator_base::M_attach_single()`, `__gnu_debug::Safe_iterator_base::M_detach()`, and `__gnu_debug::Safe_iterator_base::M_detach_single()`.

5.90.3.9 `void __gnu_debug::Safe_sequence_base::M_revalidate_singular ()` `[protected]`

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

5.90.3.10 `void __gnu_debug::Safe_sequence_base::M_swap (_Safe_sequence_base & __x)` `[protected]`,
`[noexcept]`

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.90.4 Member Data Documentation

5.90.4.1 `_Safe_iterator_base* __gnu_debug::_Safe_sequence_base::_M_const_iterators`

The list of constant iterators that reference this container.

Definition at line 193 of file `safe_base.h`.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`.

5.90.4.2 `_Safe_iterator_base* __gnu_debug::_Safe_sequence_base::_M_iterators`

The list of mutable iterators that reference this container.

Definition at line 190 of file `safe_base.h`.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`.

5.90.4.3 `unsigned int __gnu_debug::_Safe_sequence_base::_M_version` `[mutable]`

The container version number. This number may never be 0.

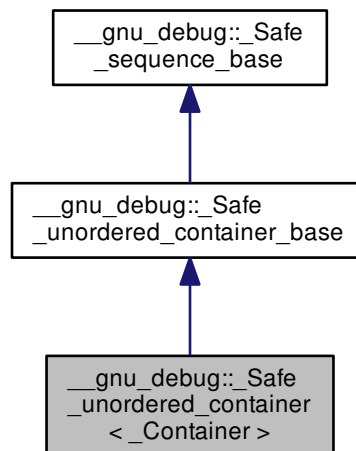
Definition at line 196 of file `safe_base.h`.

The documentation for this class was generated from the following file:

- [safe_base.h](#)

5.91 `__gnu_debug::_Safe_unordered_container<_Container>` Class Template Reference

Inheritance diagram for `__gnu_debug::_Safe_unordered_container<_Container>`:



Public Member Functions

- `void _M_attach (_Safe_iterator_base * __it, bool __constant)`
- `void _M_attach_local (_Safe_iterator_base * __it, bool __constant)`
- `void _M_attach_local_single (_Safe_iterator_base * __it, bool __constant) throw ()`
- `void _M_attach_single (_Safe_iterator_base * __it, bool __constant) throw ()`
- `void _M_detach (_Safe_iterator_base * __it)`
- `void _M_detach_local (_Safe_iterator_base * __it)`
- `void _M_detach_local_single (_Safe_iterator_base * __it) throw ()`
- `void _M_detach_single (_Safe_iterator_base * __it) throw ()`
- `void _M_invalidate_all () const`

Public Attributes

- `_Safe_iterator_base * _M_const_iterators`
- `_Safe_iterator_base * _M_const_local_iterators`
- `_Safe_iterator_base * _M_iterators`
- `_Safe_iterator_base * _M_local_iterators`
- `unsigned int _M_version`

Protected Member Functions

- `void _M_detach_all ()`
- `void _M_detach_singular ()`
- `__gnu_cxx::__mutex & _M_get_mutex () throw ()`
- `void _M_invalidate_all ()`
- `template<typename _Predicate>
void _M_invalidate_if (_Predicate __pred)`
- `template<typename _Predicate>
void _M_invalidate_local_if (_Predicate __pred)`
- `void _M_invalidate_locals ()`
- `void _M_revalidate_singular ()`
- `void _M_swap (_Safe_unordered_container_base & __x) noexcept`
- `void _M_swap (_Safe_sequence_base & __x) noexcept`

5.91.1 Detailed Description

```
template<typename _Container>
class __gnu_debug::Safe_unordered_container<_Container>
```

Base class for constructing a *safe* unordered container type that tracks iterators that reference it.

The class template `_Safe_unordered_container` simplifies the construction of *safe* unordered containers that track the iterators that reference the container, so that the iterators are notified of changes in the container that may affect their operation, e.g., if the container invalidates its iterators or is destructed. This class template may only be used by deriving from it and passing the name of the derived class as its template parameter via the curiously recurring template pattern. The derived class must have `iterator` and `const_iterator` types that are instantiations of class template `_Safe_iterator` for this container and `local_iterator` and `const_local_iterator` types that are instantiations of class template `_Safe_local_iterator` for this container. Iterators will then be tracked automatically.

Definition at line 58 of file `safe_unordered_container.h`.

5.91.2 Member Function Documentation

5.91.2.1 `void __gnu_debug::_Safe_sequence_base::_M_attach (_Safe_iterator_base * __it, bool __constant)`
[inherited]

Attach an iterator to this sequence.

5.91.2.2 `void __gnu_debug::_Safe_unordered_container_base::_M_attach_local (_Safe_iterator_base * __it, bool __constant)`
[inherited]

Attach an iterator to this container.

5.91.2.3 `void __gnu_debug::_Safe_unordered_container_base::_M_attach_local_single (_Safe_iterator_base * __it, bool __constant) throw)` [inherited]

Likewise but not thread safe.

5.91.2.4 `void __gnu_debug::_Safe_sequence_base::_M_attach_single (_Safe_iterator_base * __it, bool __constant) throw)`
[inherited]

Likewise but not thread safe.

5.91.2.5 `void __gnu_debug::_Safe_sequence_base::_M_detach (_Safe_iterator_base * __it)` [inherited]

Detach an iterator from this sequence

Referenced by `__gnu_debug::_Safe_iterator<_Iterator, _Sequence>::_Safe_iterator()`.

5.91.2.6 `void __gnu_debug::_Safe_unordered_container_base::_M_detach_all ()` [protected], [inherited]

Detach all iterators, leaving them singular.

5.91.2.7 `void __gnu_debug::_Safe_unordered_container_base::_M_detach_local (_Safe_iterator_base * __it)`
[inherited]

Detach an iterator from this container

5.91.2.8 `void __gnu_debug::_Safe_unordered_container_base::_M_detach_local_single (_Safe_iterator_base * __it) throw)`
[inherited]

Likewise but not thread safe.

5.91.2.9 `void __gnu_debug::_Safe_sequence_base::_M_detach_single (_Safe_iterator_base * __it) throw)`
[inherited]

Likewise but not thread safe.

5.91.2.10 `void __gnu_debug::_Safe_sequence_base::_M_detach_singular ()` [protected],[inherited]

Detach all singular iterators.

Postcondition

for all iterators `i` attached to this sequence, `i->_M_version == _M_version`.

5.91.2.11 `__gnu_cxx::mutex& __gnu_debug::_Safe_sequence_base::_M_get_mutex () throw` [protected],[inherited]

For use in `_Safe_sequence`.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`.

5.91.2.12 `void __gnu_debug::_Safe_sequence_base::_M_invalidate_all () const` [inline],[inherited]

Invalidates all iterators.

Definition at line 248 of file `safe_base.h`.

References `__gnu_debug::_Safe_iterator_base::_M_attach()`, `__gnu_debug::_Safe_iterator_base::_M_attach_single()`, `__gnu_debug::_Safe_iterator_base::_M_detach()`, and `__gnu_debug::_Safe_iterator_base::_M_detach_single()`.

5.91.2.13 `template<typename _Container> template<typename _Predicate> void __gnu_debug::_Safe_unordered_container<_Container>::_M_invalidate_if (_Predicate __pred)` [protected]

Invalidates all iterators `x` that reference this container, are not singular, and for which `__pred(x)` returns `true`. `__pred` will be invoked with the normal iterators nested in the safe ones.

Definition at line 38 of file `safe_unordered_container.tcc`.

References `__gnu_debug::_Safe_unordered_container<_Container>::_M_invalidate_local_if()`.

5.91.2.14 `template<typename _Container> template<typename _Predicate> void __gnu_debug::_Safe_unordered_container<_Container>::_M_invalidate_local_if (_Predicate __pred)` [protected]

Invalidates all local iterators `x` that reference this container, are not singular, and for which `__pred(x)` returns `true`. `__pred` will be invoked with the normal ilocal iterators nested in the safe ones.

Definition at line 70 of file `safe_unordered_container.tcc`.

Referenced by `__gnu_debug::_Safe_unordered_container<_Container>::_M_invalidate_if()`.

5.91.2.15 `void __gnu_debug::_Safe_sequence_base::_M_revalidate_singular ()` [protected],[inherited]

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

5.91.2.16 `void __gnu_debug::Safe_unordered_container_base::M_swap (_Safe_unordered_container_base & __x)`
`[protected], [noexcept], [inherited]`

Swap this container with the given container. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.91.2.17 `void __gnu_debug::Safe_sequence_base::M_swap (_Safe_sequence_base & __x)` `[protected],`
`[noexcept], [inherited]`

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.91.3 Member Data Documentation

5.91.3.1 `_Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_const_iterators` `[inherited]`

The list of constant iterators that reference this container.

Definition at line 193 of file `safe_base.h`.

Referenced by `__gnu_debug::Safe_sequence< _Sequence >::M_transfer_from_if()`.

5.91.3.2 `_Safe_iterator_base* __gnu_debug::Safe_unordered_container_base::M_const_local_iterators` `[inherited]`

The list of constant local iterators that reference this container.

Definition at line 125 of file `safe_unordered_base.h`.

5.91.3.3 `_Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_iterators` `[inherited]`

The list of mutable iterators that reference this container.

Definition at line 190 of file `safe_base.h`.

Referenced by `__gnu_debug::Safe_sequence< _Sequence >::M_transfer_from_if()`.

5.91.3.4 `_Safe_iterator_base* __gnu_debug::Safe_unordered_container_base::M_local_iterators` `[inherited]`

The list of mutable local iterators that reference this container.

Definition at line 122 of file `safe_unordered_base.h`.

5.91.3.5 `unsigned int __gnu_debug::Safe_sequence_base::M_version` `[mutable], [inherited]`

The container version number. This number may never be 0.

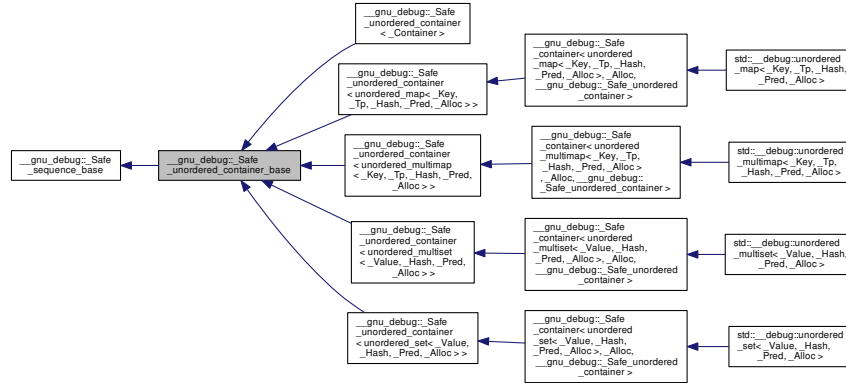
Definition at line 196 of file `safe_base.h`.

The documentation for this class was generated from the following files:

- [safe_unordered_container.h](#)
- [safe_unordered_container.tcc](#)

5.92 __gnu_debug::__Safe_unordered_container_base Class Reference

Inheritance diagram for __gnu_debug::__Safe_unordered_container_base:



Public Member Functions

- void [_M_attach](#) ([_Safe_iterator_base](#) * __it, bool __constant)
- void [_M_attach_local](#) ([_Safe_iterator_base](#) * __it, bool __constant)
- void [_M_attach_local_single](#) ([_Safe_iterator_base](#) * __it, bool __constant) throw ()
- void [_M_attach_single](#) ([_Safe_iterator_base](#) * __it, bool __constant) throw ()
- void [_M_detach](#) ([_Safe_iterator_base](#) * __it)
- void [_M_detach_local](#) ([_Safe_iterator_base](#) * __it)
- void [_M_detach_local_single](#) ([_Safe_iterator_base](#) * __it) throw ()
- void [_M_detach_single](#) ([_Safe_iterator_base](#) * __it) throw ()
- void [_M_invalidate_all](#) () const

Public Attributes

- [_Safe_iterator_base](#) * [_M_const_iterators](#)
- [_Safe_iterator_base](#) * [_M_const_local_iterators](#)
- [_Safe_iterator_base](#) * [_M_iterators](#)
- [_Safe_iterator_base](#) * [_M_local_iterators](#)
- unsigned int [_M_version](#)

Protected Member Functions

- [_Safe_unordered_container_base](#) (const [_Safe_unordered_container_base](#) &) noexcept
- [_Safe_unordered_container_base](#) ([_Safe_unordered_container_base](#) && __x) noexcept
- [~Safe_unordered_container_base](#) () noexcept
- void [_M_detach_all](#) ()
- void [_M_detach_singular](#) ()
- [__gnu_cxx::__mutex](#) & [_M_get_mutex](#) () throw ()
- void [_M_revalidate_singular](#) ()
- void [_M_swap](#) ([_Safe_unordered_container_base](#) & __x) noexcept
- void [_M_swap](#) ([_Safe_sequence_base](#) & __x) noexcept

5.92.1 Detailed Description

Base class that supports tracking of local iterators that reference an unordered container.

The `_Safe_unordered_container_base` class provides basic support for tracking iterators into an unordered container. Containers that track iterators must derived from `_Safe_unordered_container_base` publicly, so that safe iterators (which inherit `_Safe_iterator_base`) can attach to them. This class contains four linked lists of iterators, one for constant iterators, one for mutable iterators, one for constant local iterators, one for mutable local iterators and a version number that allows very fast invalidation of all iterators that reference the container.

This class must ensure that no operation on it may throw an exception, otherwise *safe* containers may fail to provide the exception-safety guarantees required by the C++ standard.

Definition at line 117 of file `safe_unordered_base.h`.

5.92.2 Constructor & Destructor Documentation

5.92.2.1 `__gnu_debug::_Safe_unordered_container_base::~_Safe_unordered_container_base () [inline], [protected], [noexcept]`

Notify all iterators that reference this container that the container is being destroyed.

Definition at line 146 of file `safe_unordered_base.h`.

5.92.3 Member Function Documentation

5.92.3.1 `void __gnu_debug::_Safe_sequence_base::M_attach (_Safe_iterator_base * __it, bool __constant) [inherited]`

Attach an iterator to this sequence.

5.92.3.2 `void __gnu_debug::_Safe_unordered_container_base::M_attach_local (_Safe_iterator_base * __it, bool __constant)`

Attach an iterator to this container.

5.92.3.3 `void __gnu_debug::_Safe_unordered_container_base::M_attach_local_single (_Safe_iterator_base * __it, bool __constant) throw)`

Likewise but not thread safe.

5.92.3.4 `void __gnu_debug::_Safe_sequence_base::M_attach_single (_Safe_iterator_base * __it, bool __constant) throw) [inherited]`

Likewise but not thread safe.

5.92.3.5 void __gnu_debug::_Safe_sequence_base::_M_detach (_Safe_iterator_base * __it) [inherited]

Detach an iterator from this sequence

Referenced by __gnu_debug::_Safe_iterator< _Iterator, _Sequence >::_Safe_iterator().

5.92.3.6 void __gnu_debug::_Safe_unordered_container_base::_M_detach_all () [protected]

Detach all iterators, leaving them singular.

5.92.3.7 void __gnu_debug::_Safe_unordered_container_base::_M_detach_local (_Safe_iterator_base * __it)

Detach an iterator from this container

5.92.3.8 void __gnu_debug::_Safe_unordered_container_base::_M_detach_local_single (_Safe_iterator_base * __it) throw)

Likewise but not thread safe.

5.92.3.9 void __gnu_debug::_Safe_sequence_base::_M_detach_single (_Safe_iterator_base * __it) throw)
[inherited]

Likewise but not thread safe.

5.92.3.10 void __gnu_debug::_Safe_sequence_base::_M_detach_singular () [protected], [inherited]

Detach all singular iterators.

Postcondition

for all iterators i attached to this sequence, i->_M_version == _M_version.

5.92.3.11 __gnu_cxx::mutex& __gnu_debug::_Safe_sequence_base::_M_get_mutex () throw) [protected],
[inherited]

For use in _Safe_sequence.

Referenced by __gnu_debug::_Safe_sequence< _Sequence >::_M_transfer_from_if().

5.92.3.12 void __gnu_debug::_Safe_sequence_base::_M_invalidate_all () const [inline], [inherited]

Invalidates all iterators.

Definition at line 248 of file safe_base.h.

References __gnu_debug::_Safe_iterator_base::_M_attach(), __gnu_debug::_Safe_iterator_base::_M_attach_single(), __gnu_debug::_Safe_iterator_base::_M_detach(), and __gnu_debug::_Safe_iterator_base::_M_detach_single().

5.92.3.13 `void __gnu_debug::Safe_sequence_base::M_revalidate_singular()` `[protected]`, `[inherited]`

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

5.92.3.14 `void __gnu_debug::Safe_unordered_container_base::M_swap(_Safe_unordered_container_base & __x)` `[protected]`, `[noexcept]`

Swap this container with the given container. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.92.3.15 `void __gnu_debug::Safe_sequence_base::M_swap(_Safe_sequence_base & __x)` `[protected]`, `[noexcept]`, `[inherited]`

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.92.4 Member Data Documentation

5.92.4.1 `_Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_const_iterators` `[inherited]`

The list of constant iterators that reference this container.

Definition at line 193 of file `safe_base.h`.

Referenced by `__gnu_debug::Safe_sequence< _Sequence >::M_transfer_from_if()`.

5.92.4.2 `_Safe_iterator_base* __gnu_debug::Safe_unordered_container_base::M_const_local_iterators`

The list of constant local iterators that reference this container.

Definition at line 125 of file `safe_unordered_base.h`.

5.92.4.3 `_Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_iterators` `[inherited]`

The list of mutable iterators that reference this container.

Definition at line 190 of file `safe_base.h`.

Referenced by `__gnu_debug::Safe_sequence< _Sequence >::M_transfer_from_if()`.

5.92.4.4 `_Safe_iterator_base* __gnu_debug::Safe_unordered_container_base::M_local_iterators`

The list of mutable local iterators that reference this container.

Definition at line 122 of file `safe_unordered_base.h`.

5.92.4.5 `unsigned int __gnu_debug::_Safe_sequence_base::_M_version` `[mutable]`, `[inherited]`

The container version number. This number may never be 0.

Definition at line 196 of file `safe_base.h`.

The documentation for this class was generated from the following file:

- [safe_unordered_base.h](#)

5.93 `__gnu_debug::_Safe_vector<_SafeSequence, _BaseSequence>` Class Template Reference

Protected Member Functions

- `_Safe_vector` (const [_Safe_vector](#) &) noexcept
- `_Safe_vector` (size_type __n) noexcept
- `_Safe_vector` ([_Safe_vector](#) &&__x) noexcept
- `bool _M_requires_reallocation` (size_type __elements) const noexcept
- `void _M_update_guaranteed_capacity` () noexcept
- `_Safe_vector` & `operator=` (const [_Safe_vector](#) &) noexcept
- `_Safe_vector` & `operator=` ([_Safe_vector](#) &&__x) noexcept

Protected Attributes

- size_type `_M_guaranteed_capacity`

5.93.1 Detailed Description

```
template<typename _SafeSequence, typename _BaseSequence>
class __gnu_debug::_Safe_vector<_SafeSequence, _BaseSequence>
```

Base class for Debug Mode vector.

Adds information about the guaranteed capacity, which is useful for detecting code which relies on non-portable implementation details of the libstdc++ reallocation policy.

Definition at line 48 of file `debug/vector`.

The documentation for this class was generated from the following file:

- [debug/vector](#)

5.94 `__gnu_debug::_Sequence_traits<_Sequence>` Struct Template Reference

Public Types

- `typedef _Distance_traits< typename _Sequence::iterator > _DistTraits`

Static Public Member Functions

- static `_DistTraits::_type _S_size` (const `_Sequence` &__seq)

5.94.1 Detailed Description

```
template<typename _Sequence>
struct __gnu_debug::_Sequence_traits< _Sequence >
```

Sequence traits giving the size of a container if possible.

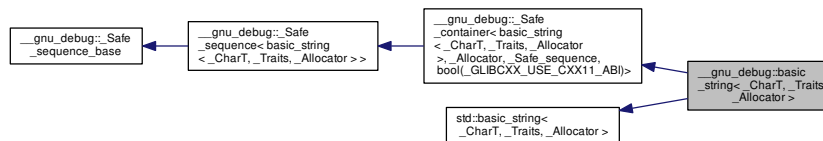
Definition at line 60 of file `safe_iterator.h`.

The documentation for this struct was generated from the following file:

- [safe_iterator.h](#)

5.95 __gnu_debug::basic_string< _CharT, _Traits, _Allocator > Class Template Reference

Inheritance diagram for `__gnu_debug::basic_string< _CharT, _Traits, _Allocator >`:



Public Types

- typedef `_Allocator` **allocator_type**
- typedef `__gnu_debug::Safe_iterator< typename _Base::const_iterator, basic_string >` **const_iterator**
- typedef `_Base::const_pointer` **const_pointer**
- typedef `_Base::const_reference` **const_reference**
- typedef `std::reverse_iterator< const_iterator >` **const_reverse_iterator**
- typedef `_Base::difference_type` **difference_type**
- typedef `__gnu_debug::Safe_iterator< typename _Base::iterator, basic_string >` **iterator**
- typedef `_Base::pointer` **pointer**
- typedef `_Base::reference` **reference**
- typedef `std::reverse_iterator< iterator >` **reverse_iterator**
- typedef `_Base::size_type` **size_type**
- typedef `_Traits` **traits_type**
- typedef `_Traits::char_type` **value_type**

Public Member Functions

- **basic_string** (const `_Allocator` &__a) noexcept
- **basic_string** (const `basic_string` &)=default
- **basic_string** (`basic_string` &&)=default
- **basic_string** (std::initializer_list< `_CharT` > __l, const `_Allocator` &__a=_Allocator())
- **basic_string** (`_Base` &&__base) noexcept
- **basic_string** (const `_Base` &__base)
- **basic_string** (const `basic_string` &__str, size_type __pos, size_type __n=_Base::npos, const `_Allocator` &__a=_Allocator())
- **basic_string** (const `_CharT` *__s, size_type __n, const `_Allocator` &__a=_Allocator())
- **basic_string** (const `_CharT` *__s, const `_Allocator` &__a=_Allocator())
- **basic_string** (size_type __n, `_CharT` __c, const `_Allocator` &__a=_Allocator())
- template<typename `_InputIterator` >
 basic_string (`_InputIterator` __begin, `_InputIterator` __end, const `_Allocator` &__a=_Allocator())
- void `_M_attach` (`_Safe_iterator_base` *__it, bool __constant)
- void `_M_attach_single` (`_Safe_iterator_base` *__it, bool __constant) throw ()
- `_Base` & `_M_base` () noexcept
- const `_Base` & `_M_base` () const noexcept
- void `_M_detach` (`_Safe_iterator_base` *__it)
- void `_M_detach_single` (`_Safe_iterator_base` *__it) throw ()
- void `_M_invalidate_all` () const
- void `_M_invalidate_if` (`_Predicate` __pred)
- void `_M_swap` (`_Safe_container` &__x) noexcept
- void `_M_transfer_from_if` (`_Safe_sequence` &__from, `_Predicate` __pred)
- `basic_string` & **append** (const `basic_string` &__str)
- `basic_string` & **append** (const `basic_string` &__str, size_type __pos, size_type __n)
- `basic_string` & **append** (const `_CharT` *__s, size_type __n)
- `basic_string` & **append** (const `_CharT` *__s)
- `basic_string` & **append** (size_type __n, `_CharT` __c)
- template<typename `_InputIterator` >
 `basic_string` & **append** (`_InputIterator` __first, `_InputIterator` __last)
- `basic_string` & **append** (const `basic_string` &__str)
- `basic_string` & **append** (const `basic_string` &__str, size_type __pos, size_type __n)
- `basic_string` & **append** (initializer_list< `_CharT` > __l)
- `basic_string` & **assign** (const `basic_string` &__x)
- `basic_string` & **assign** (`basic_string` &&__x) noexcept(noexcept(std::declval< `_Base` & >().assign(std::move(__x))))
- `basic_string` & **assign** (const `basic_string` &__str, size_type __pos, size_type __n)
- `basic_string` & **assign** (const `_CharT` *__s, size_type __n)
- `basic_string` & **assign** (const `_CharT` *__s)
- `basic_string` & **assign** (size_type __n, `_CharT` __c)
- template<typename `_InputIterator` >
 `basic_string` & **assign** (`_InputIterator` __first, `_InputIterator` __last)
- `basic_string` & **assign** (std::initializer_list< `_CharT` > __l)
- `basic_string` & **assign** (const `basic_string` &__str)
- `basic_string` & **assign** (`basic_string` &&__str)
- `basic_string` & **assign** (const `basic_string` &__str, size_type __pos, size_type __n)
- const_reference **at** (size_type __n) const
- reference **at** (size_type __n)
- reference **back** ()

- `const_reference back ()` const noexcept
- `iterator begin ()`
- `const_iterator begin ()` const noexcept
- `const _CharT * c_str ()` const noexcept
- `size_type capacity ()` const noexcept
- `const_iterator cbegin ()` const noexcept
- `const_iterator cend ()` const noexcept
- `void clear ()`
- `int compare (const basic_string &__str) const`
- `int compare (size_type __pos1, size_type __n1, const basic_string &__str) const`
- `int compare (size_type __pos1, size_type __n1, const basic_string &__str, size_type __pos2, size_type __n2) const`
- `int compare (const _CharT *__s) const`
- `int compare (size_type __pos1, size_type __n1, const _CharT *__s) const`
- `int compare (size_type __pos1, size_type __n1, const _CharT *__s, size_type __n2) const`
- `int compare (const basic_string &__str) const`
- `int compare (size_type __pos, size_type __n, const basic_string &__str) const`
- `int compare (size_type __pos1, size_type __n1, const basic_string &__str, size_type __pos2, size_type __n2) const`
- `size_type copy (_CharT *__s, size_type __n, size_type __pos=0) const`
- `const_reverse_iterator crbegin ()` const noexcept
- `const_reverse_iterator crend ()` const noexcept
- `const _CharT * data ()` const noexcept
- `bool empty ()` const noexcept
- `iterator end ()`
- `const_iterator end ()` const noexcept
- `basic_string & erase (size_type __pos=0, size_type __n=Base::npos)`
- `iterator erase (iterator __position)`
- `iterator erase (iterator __first, iterator __last)`
- `iterator erase (iterator __position)`
- `iterator erase (iterator __first, iterator __last)`
- `size_type find (const basic_string &__str, size_type __pos=0) const noexcept`
- `size_type find (const _CharT *__s, size_type __pos, size_type __n) const`
- `size_type find (const _CharT *__s, size_type __pos=0) const`
- `size_type find (_CharT __c, size_type __pos=0) const noexcept`
- `size_type find (const basic_string &__str, size_type __pos=0) const noexcept`
- `size_type find_first_not_of (const basic_string &__str, size_type __pos=0) const noexcept`
- `size_type find_first_not_of (const _CharT *__s, size_type __pos, size_type __n) const`
- `size_type find_first_not_of (const _CharT *__s, size_type __pos=0) const`
- `size_type find_first_not_of (_CharT __c, size_type __pos=0) const noexcept`
- `size_type find_first_not_of (const basic_string &__str, size_type __pos=0) const noexcept`
- `size_type find_first_of (const basic_string &__str, size_type __pos=0) const noexcept`
- `size_type find_first_of (const _CharT *__s, size_type __pos, size_type __n) const`
- `size_type find_first_of (const _CharT *__s, size_type __pos=0) const`
- `size_type find_first_of (_CharT __c, size_type __pos=0) const noexcept`
- `size_type find_first_of (const basic_string &__str, size_type __pos=0) const noexcept`
- `size_type find_last_not_of (const basic_string &__str, size_type __pos=Base::npos) const noexcept`
- `size_type find_last_not_of (const _CharT *__s, size_type __pos, size_type __n) const`
- `size_type find_last_not_of (const _CharT *__s, size_type __pos=Base::npos) const`
- `size_type find_last_not_of (_CharT __c, size_type __pos=Base::npos) const noexcept`
- `size_type find_last_not_of (const basic_string &__str, size_type __pos=npos) const noexcept`

- `size_type find_last_of` (const `basic_string` &__str, size_type __pos=`_Base::npos`) const noexcept
- `size_type find_last_of` (const `_CharT *`__s, size_type __pos, size_type __n) const
- `size_type find_last_of` (const `_CharT *`__s, size_type __pos=`_Base::npos`) const
- `size_type find_last_of` (`_CharT` __c, size_type __pos=`_Base::npos`) const noexcept
- `size_type find_last_of` (const `basic_string` &__str, size_type __pos=`npos`) const noexcept
- reference `front` ()
- const_reference `front` () const noexcept
- allocator_type `get_allocator` () const noexcept
- `basic_string` & `insert` (size_type __pos1, const `basic_string` &__str)
- `basic_string` & `insert` (size_type __pos1, const `basic_string` &__str, size_type __pos2, size_type __n)
- `basic_string` & `insert` (size_type __pos, const `_CharT *`__s, size_type __n)
- `basic_string` & `insert` (size_type __pos, const `_CharT *`__s)
- `basic_string` & `insert` (size_type __pos, size_type __n, `_CharT` __c)
- iterator `insert` (iterator __p, `_CharT` __c)
- void `insert` (iterator __p, size_type __n, `_CharT` __c)
- template<typename _InputIterator >
void `insert` (iterator __p, _InputIterator __first, _InputIterator __last)
- void `insert` (iterator __p, `std::initializer_list`< `_CharT` > __l)
- void `insert` (iterator __p, size_type __n, `_CharT` __c)
- void `insert` (iterator __p, _InputIterator __beg, _InputIterator __end)
- void `insert` (iterator __p, `initializer_list`< `_CharT` > __l)
- `basic_string` & `insert` (size_type __pos1, const `basic_string` &__str)
- `basic_string` & `insert` (size_type __pos1, const `basic_string` &__str, size_type __pos2, size_type __n)
- iterator `insert` (iterator __p, `_CharT` __c)
- size_type `length` () const noexcept
- size_type `max_size` () const noexcept
- `basic_string` & `operator+=` (const `basic_string` &__str)
- `basic_string` & `operator+=` (const `_CharT *`__s)
- `basic_string` & `operator+=` (`_CharT` __c)
- `basic_string` & `operator+=` (`std::initializer_list`< `_CharT` > __l)
- `basic_string` & `operator+=` (const `basic_string` &__str)
- `basic_string` & `operator=` (const `basic_string` &)=default
- `basic_string` & `operator=` (`basic_string` &&)=default
- `basic_string` & `operator=` (const `_CharT *`__s)
- `basic_string` & `operator=` (`_CharT` __c)
- `basic_string` & `operator=` (`std::initializer_list`< `_CharT` > __l)
- const_reference `operator[]` (size_type __pos) const noexcept
- reference `operator[]` (size_type __pos)
- void `pop_back` ()
- void `push_back` (`_CharT` __c)
- `reverse_iterator` `rbegin` ()
- const_reverse_iterator `rbegin` () const noexcept
- `reverse_iterator` `rend` ()
- const_reverse_iterator `rend` () const noexcept
- `basic_string` & `replace` (size_type __pos1, size_type __n1, const `basic_string` &__str)
- `basic_string` & `replace` (size_type __pos1, size_type __n1, const `basic_string` &__str, size_type __pos2, size_type __n2)
- `basic_string` & `replace` (size_type __pos, size_type __n1, const `_CharT *`__s, size_type __n2)
- `basic_string` & `replace` (size_type __pos, size_type __n1, const `_CharT *`__s)
- `basic_string` & `replace` (size_type __pos, size_type __n1, size_type __n2, `_CharT` __c)
- `basic_string` & `replace` (iterator __i1, iterator __i2, const `basic_string` &__str)

- [basic_string](#) & [replace](#) ([iterator](#) __i1, [iterator](#) __i2, const [_CharT](#) *__s, [size_type](#) __n)
- [basic_string](#) & [replace](#) ([iterator](#) __i1, [iterator](#) __i2, const [_CharT](#) *__s)
- [basic_string](#) & [replace](#) ([iterator](#) __i1, [iterator](#) __i2, [size_type](#) __n, [_CharT](#) __c)
- [template](#)<[typename](#) [_InputIterator](#) >
[basic_string](#) & [replace](#) ([iterator](#) __i1, [iterator](#) __i2, [_InputIterator](#) __j1, [_InputIterator](#) __j2)
- [basic_string](#) & [replace](#) ([iterator](#) __i1, [iterator](#) __i2, [std::initializer_list](#)< [_CharT](#) > __l)
- [basic_string](#) & [replace](#) ([size_type](#) __pos, [size_type](#) __n, const [basic_string](#) & __str)
- [basic_string](#) & [replace](#) ([size_type](#) __pos1, [size_type](#) __n1, const [basic_string](#) & __str, [size_type](#) __pos2, [size_type](#) __n2)
- [basic_string](#) & [replace](#) ([iterator](#) __i1, [iterator](#) __i2, const [basic_string](#) & __str)
- [basic_string](#) & [replace](#) ([iterator](#) __i1, [iterator](#) __i2, const [_CharT](#) *__s, [size_type](#) __n)
- [basic_string](#) & [replace](#) ([iterator](#) __i1, [iterator](#) __i2, const [_CharT](#) *__s)
- [basic_string](#) & [replace](#) ([iterator](#) __i1, [iterator](#) __i2, [size_type](#) __n, [_CharT](#) __c)
- [basic_string](#) & [replace](#) ([iterator](#) __i1, [iterator](#) __i2, [_InputIterator](#) __k1, [_InputIterator](#) __k2)
- [basic_string](#) & [replace](#) ([iterator](#) __i1, [iterator](#) __i2, [_CharT](#) *__k1, [_CharT](#) *__k2)
- [basic_string](#) & [replace](#) ([iterator](#) __i1, [iterator](#) __i2, const [_CharT](#) *__k1, const [_CharT](#) *__k2)
- [basic_string](#) & [replace](#) ([iterator](#) __i1, [iterator](#) __i2, [iterator](#) __k1, [iterator](#) __k2)
- [basic_string](#) & [replace](#) ([iterator](#) __i1, [iterator](#) __i2, const [iterator](#) __k1, const [iterator](#) __k2)
- [basic_string](#) & [replace](#) ([iterator](#) __i1, [iterator](#) __i2, [initializer_list](#)< [_CharT](#) > __l)
- void [reserve](#) ([size_type](#) __res_arg=0)
- void [resize](#) ([size_type](#) __n, [_CharT](#) __c)
- void [resize](#) ([size_type](#) __n)
- [size_type](#) [rfind](#) (const [basic_string](#) & __str, [size_type](#) __pos=[_Base::npos](#)) const noexcept
- [size_type](#) [rfind](#) (const [_CharT](#) *__s, [size_type](#) __pos, [size_type](#) __n) const
- [size_type](#) [rfind](#) (const [_CharT](#) *__s, [size_type](#) __pos=[_Base::npos](#)) const
- [size_type](#) [rfind](#) ([_CharT](#) __c, [size_type](#) __pos=[_Base::npos](#)) const noexcept
- [size_type](#) [rfind](#) (const [basic_string](#) & __str, [size_type](#) __pos=[npos](#)) const noexcept
- void [shrink_to_fit](#) () noexcept
- [size_type](#) [size](#) () const noexcept
- [basic_string](#) [substr](#) ([size_type](#) __pos=0, [size_type](#) __n=[_Base::npos](#)) const
- void [swap](#) ([basic_string](#) & __x) noexcept(*/*conditional */*)
- void [swap](#) ([basic_string](#) & __s)

Public Attributes

- [_Safe_iterator_base](#) * [_M_const_iterators](#)
- [_Safe_iterator_base](#) * [_M_iterators](#)
- unsigned int [_M_version](#)

Static Public Attributes

- static const [size_type](#) [npos](#)

Protected Member Functions

- void [_M_detach_all](#) ()
- void [_M_detach_singular](#) ()
- [__gnu_cxx::__mutex](#) & [_M_get_mutex](#) () throw ()
- void [_M_revalidate_singular](#) ()
- [_Safe_container](#) & [_M_safe](#) () noexcept
- void [_M_swap](#) ([_Safe_sequence_base](#) & __x) noexcept

5.95.1 Detailed Description

```
template<typename _CharT, typename _Traits = std::char_traits<_CharT>, typename _Allocator = std::allocator<_CharT>>
class __gnu_debug::basic_string<_CharT, _Traits, _Allocator>
```

Class `std::basic_string` with safety/checking/debug instrumentation.

Definition at line 42 of file `debug/string`.

5.95.2 Member Function Documentation

5.95.2.1 `void __gnu_debug::_Safe_sequence_base::_M_attach (_Safe_iterator_base * __it, bool __constant)`
[*inherited*]

Attach an iterator to this sequence.

5.95.2.2 `void __gnu_debug::_Safe_sequence_base::_M_attach_single (_Safe_iterator_base * __it, bool __constant) throw)`
[*inherited*]

Likewise but not thread safe.

5.95.2.3 `void __gnu_debug::_Safe_sequence_base::_M_detach (_Safe_iterator_base * __it)` [*inherited*]

Detach an iterator from this sequence

Referenced by `__gnu_debug::_Safe_iterator<_Iterator, _Sequence>::_Safe_iterator()`.

5.95.2.4 `void __gnu_debug::_Safe_sequence_base::_M_detach_all ()` [*protected*], [*inherited*]

Detach all iterators, leaving them singular.

5.95.2.5 `void __gnu_debug::_Safe_sequence_base::_M_detach_single (_Safe_iterator_base * __it) throw)`
[*inherited*]

Likewise but not thread safe.

5.95.2.6 `void __gnu_debug::_Safe_sequence_base::_M_detach_singular ()` [*protected*], [*inherited*]

Detach all singular iterators.

Postcondition

for all iterators `i` attached to this sequence, `i->_M_version == _M_version`.

5.95.2.7 `__gnu_cxx::mutex& __gnu_debug::Safe_sequence_base::M_get_mutex () throw` `[protected]`,
`[inherited]`

For use in `_Safe_sequence`.

Referenced by `__gnu_debug::Safe_sequence<_Sequence>::M_transfer_from_if()`.

5.95.2.8 `void __gnu_debug::Safe_sequence_base::M_invalidate_all () const` `[inline]`, `[inherited]`

Invalidates all iterators.

Definition at line 248 of file `safe_base.h`.

References `__gnu_debug::Safe_iterator_base::M_attach()`, `__gnu_debug::Safe_iterator_base::M_attach_single()`, `__gnu_debug::Safe_iterator_base::M_detach()`, and `__gnu_debug::Safe_iterator_base::M_detach_single()`.

5.95.2.9 `void __gnu_debug::Safe_sequence<basic_string<_CharT, _Traits, _Allocator>>::M_invalidate_if (`
`_Predicate __pred)` `[inherited]`

Invalidates all iterators `x` that reference this sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

5.95.2.10 `void __gnu_debug::Safe_sequence_base::M_revalidate_singular ()` `[protected]`, `[inherited]`

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

5.95.2.11 `void __gnu_debug::Safe_sequence_base::M_swap (_Safe_sequence_base & __x)` `[protected]`,
`[noexcept]`, `[inherited]`

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.95.2.12 `void __gnu_debug::Safe_sequence<basic_string<_CharT, _Traits, _Allocator>>::M_transfer_from_if`
`(_Safe_sequence<basic_string<_CharT, _Traits, _Allocator>> & __from, _Predicate __pred)`
`[inherited]`

Transfers all iterators `x` that reference `from` sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

5.95.2.13 `basic_string& std::basic_string<_CharT, _Traits, _Allocator>::append (const basic_string<_CharT, _Traits,`
`_Allocator> & __str)` `[inherited]`

Append a string to this string.

Parameters

<code>__str</code>	The string to append.
--------------------	-----------------------

Returns

Reference to this string.

5.95.2.14 `basic_string& std::basic_string<_CharT, _Traits, _Allocator>::append (const basic_string<_CharT, _Traits, _Allocator> &__str, size_type __pos, size_type __n)` `[inherited]`

Append a substring.

Parameters

<code>__str</code>	The string to append.
<code>__pos</code>	Index of the first character of <code>str</code> to append.
<code>__n</code>	The number of characters to append.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	if <code>__pos</code> is not a valid index.
--------------------------------	---

This function appends `__n` characters from `__str` starting at `__pos` to this string. If `__n` is larger than the number of available characters in `__str`, the remainder of `__str` is appended.

5.95.2.15 `basic_string& std::basic_string<_CharT, _Traits, _Allocator>::append (initializer_list<_CharT> __l)` `[inline], [inherited]`

Append an `initializer_list` of characters.

Parameters

<code>__l</code>	The <code>initializer_list</code> of characters to append.
------------------	--

Returns

Reference to this string.

Definition at line 3576 of file `basic_string.h`.

5.95.2.16 `basic_string& std::basic_string<_CharT, _Traits, _Allocator>::assign (const basic_string<_CharT, _Traits, _Allocator> &__str)` [inherited]

Set value to contents of another string.

Parameters

<code>__str</code>	Source string to use.
--------------------	-----------------------

Returns

Reference to this string.

5.95.2.17 `basic_string& std::basic_string<_CharT, _Traits, _Allocator>::assign (basic_string<_CharT, _Traits, _Allocator> && __str)` `[inline]`, `[inherited]`

Set value to contents of another string.

Parameters

<code>__str</code>	Source string to use.
--------------------	-----------------------

Returns

Reference to this string.

This function sets this string to the exact contents of `__str`. `__str` is a valid, but unspecified string.

Definition at line 3626 of file `basic_string.h`.

5.95.2.18 `basic_string& std::basic_string<_CharT, _Traits, _Allocator>::assign (const basic_string<_CharT, _Traits, _Allocator> & __str, size_type __pos, size_type __n)` `[inline]`, `[inherited]`

Set value to a substring of a string.

Parameters

<code>__str</code>	The string to use.
<code>__pos</code>	Index of the first character of str.
<code>__n</code>	Number of characters to use.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	if <code>pos</code> is not a valid index.
--------------------------------	---

This function sets this string to the substring of `__str` consisting of `__n` characters at `__pos`. If `__n` is larger than the number of available characters in `__str`, the remainder of `__str` is used.

Definition at line 3647 of file `basic_string.h`.

5.95.2.19 `const_reference std::basic_string<_CharT, _Traits, _Allocator>::at (size_type __n) const` `[inline]`,
`[inherited]`

Provides access to the data contained in the string.

Parameters

<code>__pos</code>	The index of the character to access.
<code>__n</code>	

Returns

Read-only (const) reference to the character.

Exceptions

<code>std::out_of_range</code>	If <code>n</code> is an invalid index.
--------------------------------	--

This function provides for safer data access. The parameter is first checked that it is in the range of the string. The function throws `out_of_range` if the check fails.

Definition at line 3392 of file `basic_string.h`.

5.95.2.20 `reference std::basic_string<_CharT, _Traits, _Allocator>::at (size_type __n)` `[inline]`, `[inherited]`

Provides access to the data contained in the string.

Parameters

<code>__pos</code>	The index of the character to access.
<code>__n</code>	

Returns

Read/write reference to the character.

Exceptions

<code>std::out_of_range</code>	If <code>n</code> is an invalid index.
--------------------------------	--

This function provides for safer data access. The parameter is first checked that it is in the range of the string. The

function throws `out_of_range` if the check fails. Success results in unsharing the string.

Definition at line 3414 of file `basic_string.h`.

5.95.2.21 `reference std::basic_string<_CharT, _Traits, _Allocator>::back ()` `[inline]`, `[inherited]`

Returns a read/write reference to the data at the last element of the string.

Definition at line 3453 of file `basic_string.h`.

5.95.2.22 `const_reference std::basic_string<_CharT, _Traits, _Allocator>::back () const` `[inline]`, `[noexcept]`, `[inherited]`

Returns a read-only (constant) reference to the data at the last element of the string.

Definition at line 3464 of file `basic_string.h`.

5.95.2.23 `size_type std::basic_string<_CharT, _Traits, _Allocator>::capacity () const` `[inline]`, `[noexcept]`, `[inherited]`

Returns the total number of characters that the string can hold before needing to allocate more memory.

Definition at line 3302 of file `basic_string.h`.

5.95.2.24 `int std::basic_string<_CharT, _Traits, _Allocator>::compare (const basic_string<_CharT, _Traits, _Allocator> &__str) const` `[inline]`, `[inherited]`

Compare to a string.

Parameters

<code>__str</code>	String to compare against.
--------------------	----------------------------

Returns

Integer < 0 , 0 , or > 0 .

Returns an integer < 0 if this string is ordered before `__str`, 0 if their values are equivalent, or > 0 if this string is ordered after `__str`. Determines the effective length `rlen` of the strings to compare as the smallest of `size()` and `str.size()`. The function then compares the two strings by calling `traits::compare(data(), str.data(), rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

Definition at line 4776 of file `basic_string.h`.

5.95.2.25 `int std::basic_string<_CharT, _Traits, _Allocator>::compare (size_type __pos, size_type __n, const basic_string<_CharT, _Traits, _Allocator> &__str) const` `[inherited]`

Compare substring to a string.

Parameters

<code>__pos</code>	Index of first character of substring.
<code>__n</code>	Number of characters in substring.
<code>__str</code>	String to compare against.

Returns

Integer < 0, 0, or > 0.

Form the substring of this string from the `__n` characters starting at `__pos`. Returns an integer < 0 if the substring is ordered before `__str`, 0 if their values are equivalent, or > 0 if the substring is ordered after `__str`. Determines the effective length `rlen` of the strings to compare as the smallest of the length of the substring and `__str.size()`. The function then compares the two strings by calling `traits::compare(substring.data(),str.data(),rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

5.95.2.26 `int std::basic_string<_CharT,_Traits,_Allocator>::compare (size_type __pos1, size_type __n1, const basic_string<_CharT,_Traits,_Allocator> & __str, size_type __pos2, size_type __n2) const` `[inherited]`

Compare substring to a substring.

Parameters

<code>__pos1</code>	Index of first character of substring.
<code>__n1</code>	Number of characters in substring.
<code>__str</code>	String to compare against.
<code>__pos2</code>	Index of first character of substring of <code>str</code> .
<code>__n2</code>	Number of characters in substring of <code>str</code> .

Returns

Integer < 0, 0, or > 0.

Form the substring of this string from the `__n1` characters starting at `__pos1`. Form the substring of `__str` from the `__n2` characters starting at `__pos2`. Returns an integer < 0 if this substring is ordered before the substring of `__str`, 0 if their values are equivalent, or > 0 if this substring is ordered after the substring of `__str`. Determines the effective length `rlen` of the strings to compare as the smallest of the lengths of the substrings. The function then compares the two strings by calling `traits::compare(substring.data(),str.substr(pos2,n2).data(),rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

5.95.2.27 `bool std::basic_string<_CharT,_Traits,_Allocator>::empty () const` `[inline],[noexcept],`
`[inherited]`

Returns true if the string is empty. Equivalent to `*this == ""`.

Definition at line 3338 of file `basic_string.h`.

5.95.2.28 `iterator std::basic_string<_CharT,_Traits,_Allocator>::erase (iterator __position)` `[inline],`
`[inherited]`

Remove one character.

Parameters

<code>__position</code>	Iterator referencing the character to remove.
-------------------------	---

Returns

iterator referencing same location after removal.

Removes the character at `__position` from this string. The value of the string doesn't change if an error is thrown.

Definition at line 3925 of file `basic_string.h`.

5.95.2.29 `iterator std::basic_string<_CharT, _Traits, _Allocator>::erase (iterator __first, iterator __last)`
`[inherited]`

Remove a range of characters.

Parameters

<code>__first</code>	Iterator referencing the first character to remove.
<code>__last</code>	Iterator referencing the end of the range.

Returns

Iterator referencing location of first after removal.

Removes the characters in the range `[first,last)` from this string. The value of the string doesn't change if an error is thrown.

5.95.2.30 `size_type std::basic_string<_CharT, _Traits, _Allocator>::find (const basic_string<_CharT, _Traits, _Allocator> & __str, size_type __pos = 0) const` `[inline], [noexcept], [inherited]`

Find position of a string.

Parameters

<code>__str</code>	String to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of start of first occurrence.

Starting from `__pos`, searches forward for value of `__str` within this string. If found, returns the index where it begins. If not found, returns `npos`.

Definition at line 4397 of file `basic_string.h`.

5.95.2.31 `size_type std::basic_string<_CharT, _Traits, _Allocator>::find_first_not_of (const basic_string<_CharT, _Traits, _Allocator> & __str, size_type __pos = 0) const` [inline], [noexcept], [inherited]

Find position of a character not in string.

Parameters

<code>__str</code>	String containing characters to avoid.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for a character not contained in `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 4630 of file `basic_string.h`.

5.95.2.32 `size_type std::basic_string<_CharT, _Traits, _Allocator>::find_first_of (const basic_string<_CharT, _Traits, _Allocator> & __str, size_type __pos = 0) const` [inline], [noexcept], [inherited]

Find position of a character of string.

Parameters

<code>__str</code>	String containing characters to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for one of the characters of `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 4503 of file `basic_string.h`.

5.95.2.33 `size_type std::basic_string<_CharT, _Traits, _Allocator>::find_last_not_of (const basic_string<_CharT, _Traits, _Allocator> & __str, size_type __pos = npos) const` [inline], [noexcept], [inherited]

Find last position of a character not in string.

Parameters

<code>__str</code>	String containing characters to avoid.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for a character not contained in `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 4693 of file `basic_string.h`.

5.95.2.34 `size_type std::basic_string<_CharT, _Traits, _Allocator>::find_last_of (const basic_string<_CharT, _Traits, _Allocator> & __str, size_type __pos = npos) const` `[inline]`, `[noexcept]`, `[inherited]`

Find last position of a character of string.

Parameters

<code>__str</code>	String containing characters to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for one of the characters of `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 4567 of file `basic_string.h`.

5.95.2.35 `reference std::basic_string<_CharT, _Traits, _Allocator>::front ()` `[inline]`, `[inherited]`

Returns a read/write reference to the data at the first element of the string.

Definition at line 3431 of file `basic_string.h`.

5.95.2.36 `const_reference std::basic_string<_CharT, _Traits, _Allocator>::front () const` `[inline]`, `[noexcept]`, `[inherited]`

Returns a read-only (constant) reference to the data at the first element of the string.

Definition at line 3442 of file `basic_string.h`.

5.95.2.37 `allocator_type std::basic_string<_CharT, _Traits, _Allocator>::get_allocator () const` `[inline]`, `[noexcept]`, `[inherited]`

Return copy of allocator used to construct this string.

Definition at line 4368 of file `basic_string.h`.

5.95.2.38 `void std::basic_string<_CharT, _Traits, _Allocator>::insert (iterator __p, size_type __n, _CharT __c)` `[inline]`, `[inherited]`

Insert multiple characters.

Parameters

<code>__p</code>	Iterator referencing location in string to insert at.
<code>__n</code>	Number of characters to insert
<code>__c</code>	The character to insert.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts `__n` copies of character `__c` starting at the position referenced by iterator `__p`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 3732 of file `basic_string.h`.

```
5.95.2.39 void std::basic_string<_CharT, _Traits, _Allocator>::insert ( iterator __p, _InputIterator __beg, _InputIterator __end
) [inline],[inherited]
```

Insert a range of characters.

Parameters

<code>__p</code>	Iterator referencing location in string to insert at.
<code>__beg</code>	Start of range.
<code>__end</code>	End of range.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts characters in range `[__beg, __end)`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 3749 of file `basic_string.h`.

```
5.95.2.40 void std::basic_string<_CharT, _Traits, _Allocator>::insert ( iterator __p, initializer_list<_CharT> __l )
[inline],[inherited]
```

Insert an `initializer_list` of characters.

Parameters

<code>__p</code>	Iterator referencing location in string to insert at.
<code>__l</code>	The <code>initializer_list</code> of characters to insert.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Definition at line 3760 of file `basic_string.h`.

5.95.2.41 `basic_string& std::basic_string<_CharT, _Traits, _Allocator>::insert (size_type __pos1, const basic_string<_CharT, _Traits, _Allocator> & __str)` `[inline]`, `[inherited]`

Insert value of a string.

Parameters

<code>__pos1</code>	Iterator referencing location in string to insert at.
<code>__str</code>	The string to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts value of `__str` starting at `__pos1`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 3780 of file `basic_string.h`.

5.95.2.42 `basic_string& std::basic_string<_CharT, _Traits, _Allocator>::insert (size_type __pos1, const basic_string<_CharT, _Traits, _Allocator> & __str, size_type __pos2, size_type __n)` `[inline]`, `[inherited]`

Insert a substring.

Parameters

<code>__pos1</code>	Iterator referencing location in string to insert at.
<code>__str</code>	The string to insert.
<code>__pos2</code>	Start of characters in <code>str</code> to insert.
<code>__n</code>	Number of characters to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
<code>std::out_of_range</code>	If <code>pos1 > size()</code> or <code>__pos2 > str.size()</code> .

Starting at `pos1`, insert `__n` character of `__str` beginning with `__pos2`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. If `__pos1` is beyond the end of this string or `__pos2` is beyond the end of `__str`, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 3802 of file `basic_string.h`.

5.95.2.43 `iterator std::basic_string<_CharT, _Traits, _Allocator>::insert (iterator __p, _CharT __c)` `[inline]`,
`[inherited]`

Insert one character.

Parameters

<code>__p</code>	Iterator referencing position in string to insert at.
<code>__c</code>	The character to insert.

Returns

Iterator referencing newly inserted char.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts character `__c` at position referenced by `__p`. If adding character causes the length to exceed `max_size()`, `length_error` is thrown. If `__p` is beyond end of string, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 3884 of file `basic_string.h`.

5.95.2.44 `size_type std::basic_string<_CharT, _Traits, _Allocator>::length () const` `[inline]`, `[noexcept]`,
`[inherited]`

Returns the number of characters in the string, not including any null-termination.

Definition at line 3245 of file `basic_string.h`.

5.95.2.45 `size_type std::basic_string<_CharT, _Traits, _Allocator>::max_size () const` `[inline]`, `[noexcept]`,
`[inherited]`

Returns the `size()` of the largest possible string.

Definition at line 3250 of file `basic_string.h`.

5.95.2.46 `basic_string& std::basic_string<_CharT, _Traits, _Allocator>::operator+=(const basic_string<_CharT, _Traits, _Allocator> &__str)` `[inline]`, `[inherited]`

Append a string to this string.

Parameters

<code>__str</code>	The string to append.
--------------------	-----------------------

Returns

Reference to this string.

Definition at line 3478 of file `basic_string.h`.

5.95.2.47 `basic_string& std::basic_string<_CharT, _Traits, _Allocator>::replace (size_type __pos, size_type __n, const basic_string<_CharT, _Traits, _Allocator> &__str)` `[inline]`, `[inherited]`

Replace characters with value from another string.

Parameters

<code>__pos</code>	Index of first character to replace.
<code>__n</code>	Number of characters to be replaced.
<code>__str</code>	String to insert.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <code>pos</code> is beyond the end of this string.
<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range `[__pos, __pos+__n)` from this string. In place, the value of `__str` is inserted. If `__pos` is beyond end of string, `out_of_range` is thrown. If the length of the result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 3979 of file `basic_string.h`.

5.95.2.48 `basic_string& std::basic_string<_CharT, _Traits, _Allocator>::replace (size_type __pos1, size_type __n1, const basic_string<_CharT, _Traits, _Allocator> &__str, size_type __pos2, size_type __n2)` `[inline]`, `[inherited]`

Replace characters with value from another string.

Parameters

<code>__pos1</code>	Index of first character to replace.
<code>__n1</code>	Number of characters to be replaced.
<code>__str</code>	String to insert.
<code>__pos2</code>	Index of first character of str to use.
<code>__n2</code>	Number of characters from str to use.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <code>__pos1 > size()</code> or <code>__pos2 > __str.size()</code> .
<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range `[__pos1, __pos1 + n)` from this string. In place, the value of `__str` is inserted. If `__pos` is beyond end of string, `out_of_range` is thrown. If the length of the result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 4001 of file `basic_string.h`.

5.95.2.49 `basic_string& std::basic_string<_CharT, _Traits, _Allocator>::replace (iterator __i1, iterator __i2, const basic_string<_CharT, _Traits, _Allocator> &__str)` `[inline]`, `[inherited]`

Replace range of characters with string.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__str</code>	String value to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[__i1, __i2)`. In place, the value of `__str` is inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 4088 of file `basic_string.h`.

5.95.2.50 `basic_string& std::basic_string<_CharT, _Traits, _Allocator>::replace (iterator __i1, iterator __i2, const _CharT* __s, size_type __n)` `[inline]`, `[inherited]`

Replace range of characters with C substring.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__s</code>	C string value to insert.
<code>__n</code>	Number of characters from s to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[__i1, __i2)`. In place, the first `__n` characters of `__s` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 4107 of file `basic_string.h`.

5.95.2.51 `basic_string& std::basic_string<_CharT, _Traits, _Allocator>::replace (iterator __i1, iterator __i2, const _CharT* __s)` `[inline]`, `[inherited]`

Replace range of characters with C string.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__s</code>	C string value to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[__i1,__i2)`. In place, the characters of `__s` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 4128 of file `basic_string.h`.

5.95.2.52 `basic_string& std::basic_string<_CharT, _Traits, _Allocator>::replace (iterator __i1, iterator __i2, size_type __n, _CharT __c)` `[inline]`, `[inherited]`

Replace range of characters with multiple characters.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__n</code>	Number of characters to insert.
<code>__c</code>	Character to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[__i1,__i2)`. In place, `__n` copies of `__c` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 4149 of file `basic_string.h`.

5.95.2.53 `basic_string& std::basic_string<_CharT, _Traits, _Allocator>::replace (iterator __i1, iterator __i2, _InputIterator __k1, _InputIterator __k2)` `[inline]`, `[inherited]`

Replace range of characters with range.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__k1</code>	Iterator referencing start of range to insert.
<code>__k2</code>	Iterator referencing end of range to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[__i1, __i2)`. In place, characters in the range `[__k1, __k2)` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 4173 of file `basic_string.h`.

5.95.254 `basic_string& std::basic_string<_CharT, _Traits, _Allocator>::replace (iterator __i1, iterator __i2, initializer_list<_CharT> __l)` `[inline]`, `[inherited]`

Replace range of characters with `initializer_list`.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__l</code>	The <code>initializer_list</code> of characters to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[__i1, __i2)`. In place, characters in the range `[__k1, __k2)` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 4242 of file `basic_string.h`.

5.95.255 `void std::basic_string<_CharT, _Traits, _Allocator>::reserve (size_type __res_arg = 0)` `[inherited]`

Attempt to preallocate enough memory for specified number of characters.

Parameters

<code>__res_arg</code>	Number of characters required.
------------------------	--------------------------------

Exceptions

<code>std::length_error</code>	If <code>__res_arg</code> exceeds <code>max_size()</code> .
--------------------------------	---

This function attempts to reserve enough memory for the string to hold the specified number of characters. If the number requested is more than `max_size()`, `length_error` is thrown.

The advantage of this function is that if optimal code is a necessity and the user can determine the string length that will be required, the user can reserve the memory in advance, and thus prevent a possible reallocation of memory and copying of string data.

5.95.2.56 `size_type std::basic_string<_CharT, _Traits, _Allocator>::rfind (const basic_string<_CharT, _Traits, _Allocator> & __str, size_type __pos = npos) const` `[inline], [noexcept], [inherited]`

Find last position of a string.

Parameters

<code>__str</code>	String to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of start of last occurrence.

Starting from `__pos`, searches backward for value of `__str` within this string. If found, returns the index where it begins. If not found, returns `npos`.

Definition at line 4442 of file `basic_string.h`.

5.95.2.57 `size_type std::basic_string<_CharT, _Traits, _Allocator>::size () const` `[inline], [noexcept], [inherited]`

Returns the number of characters in the string, not including any null-termination.

Definition at line 3239 of file `basic_string.h`.

5.95.2.58 `void std::basic_string<_CharT, _Traits, _Allocator>::swap (basic_string<_CharT, _Traits, _Allocator> & __s)` `[inherited]`

Swap contents with another string.

Parameters

<code>__s</code>	String to swap with.
------------------	----------------------

Exchanges the contents of this string with that of `__s` in constant time.

5.95.3 Member Data Documentation

5.95.3.1 `_Safe_iterator_base* __gnu_debug::_Safe_sequence_base::M_const_iterators` `[inherited]`

The list of constant iterators that reference this container.

Definition at line 193 of file `safe_base.h`.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`.

5.95.3.2 `_Safe_iterator_base* __gnu_debug::_Safe_sequence_base::M_iterators` `[inherited]`

The list of mutable iterators that reference this container.

Definition at line 190 of file `safe_base.h`.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`.

5.95.3.3 `unsigned int __gnu_debug::_Safe_sequence_base::M_version` `[mutable], [inherited]`

The container version number. This number may never be 0.

Definition at line 196 of file `safe_base.h`.

5.95.3.4 `const size_type std::basic_string<_CharT, _Traits, _Allocator>::npos` `[static], [inherited]`

Value returned by various member functions when they fail.

Definition at line 2801 of file `basic_string.h`.

The documentation for this class was generated from the following file:

- [debug/string](#)

5.96 `__gnu_parallel::__accumulate_binop_reduct<_BinOp>` Struct Template Reference

Public Member Functions

- `__accumulate_binop_reduct` (`_BinOp` & `__b`)
- `template<typename _Result, typename _Addend>`
`_Result operator()` (`const _Result` & `__x`, `const _Addend` & `__y`)

Public Attributes

- `_BinOp` & `__binop`

5.96.1 Detailed Description

```
template<typename _BinOp>
struct __gnu_parallel::__accumulate_binop_reduct< _BinOp >
```

General reduction, using a binary operator.

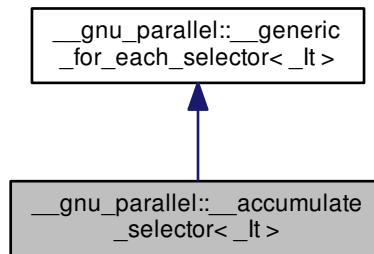
Definition at line 335 of file `for_each_selectors.h`.

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

5.97 __gnu_parallel::__accumulate_selector< _It > Struct Template Reference

Inheritance diagram for `__gnu_parallel::__accumulate_selector< _It >`:



Public Member Functions

- `template<typename _Op >`
`std::iterator_traits< _It >::value_type operator\(\) (_Op __o, _It __i)`

Public Attributes

- `_It _M_finish_iterator`

5.97.1 Detailed Description

```
template<typename _It>
struct __gnu_parallel::__accumulate_selector< _It >
```

`std::accumulate()` selector.

Definition at line 208 of file `for_each_selectors.h`.

5.97.2 Member Function Documentation

5.97.2.1 `template<typename _It> template<typename _Op> std::iterator_traits<_It>::value_type
__gnu_parallel::__accumulate_selector<_It>::operator()(_Op __o, _It __i) [inline]`

Functor execution.

Parameters

<code>__o</code>	Operator (unused).
<code>__i</code>	iterator referencing object.

Returns

The current value.

Definition at line 216 of file `for_each_selectors.h`.

5.97.3 Member Data Documentation

5.97.3.1 `template<typename _It> _It __gnu_parallel::__generic_for_each_selector<_It>::M_finish_iterator
[inherited]`

Iterator on last element processed; needed for some algorithms (e. g. `std::transform()`).

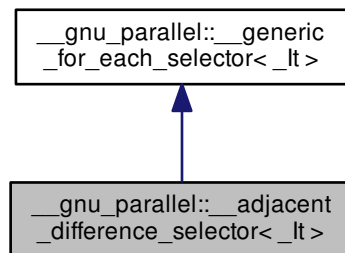
Definition at line 47 of file `for_each_selectors.h`.

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

5.98 __gnu_parallel::__adjacent_difference_selector<_It> Struct Template Reference

Inheritance diagram for `__gnu_parallel::__adjacent_difference_selector<_It>`:



Public Member Functions

- `template<typename _Op >`
`bool operator() (_Op &__o, _It __i)`

Public Attributes

- `_It _M_finish_iterator`

5.98.1 Detailed Description

```
template<typename _It>
struct __gnu_parallel::__adjacent_difference_selector< _It >
```

Selector that returns the difference between two adjacent __elements.

Definition at line 269 of file `for_each_selectors.h`.

5.98.2 Member Data Documentation

5.98.2.1 `template<typename _It > _It __gnu_parallel::__generic_for_each_selector< _It >::__M_finish_iterator`
[inherited]

__Iterator on last element processed; needed for some algorithms (e. g. `std::transform()`).

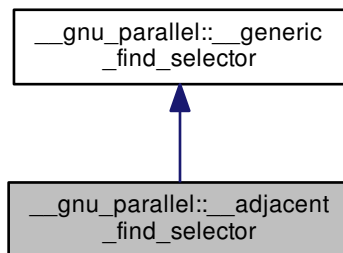
Definition at line 47 of file `for_each_selectors.h`.

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

5.99 __gnu_parallel::__adjacent_find_selector Struct Reference

Inheritance diagram for `__gnu_parallel::__adjacent_find_selector`:



Public Member Functions

- `template<typename _RAIter1, typename _RAIter2, typename _Pred >`
`std::pair<_RAIter1, _RAIter2 > _M_sequential_algorithm (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred)`
- `template<typename _RAIter1, typename _RAIter2, typename _Pred >`
`bool operator() (_RAIter1 __i1, _RAIter2 __i2, _Pred __pred)`

5.99.1 Detailed Description

Test predicate on two adjacent elements.

Definition at line 80 of file `find_selectors.h`.

5.99.2 Member Function Documentation

5.99.2.1 `template<typename _RAIter1, typename _RAIter2, typename _Pred > std::pair<_RAIter1, _RAIter2>`
`__gnu_parallel::__adjacent_find_selector::__M_sequential_algorithm (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2`
`__begin2, _Pred __pred) \[inline\]`

Corresponding sequential algorithm on a sequence.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence.
<code>__pred</code>	Find predicate.

Definition at line 105 of file `find_selectors.h`.

References `std::make_pair()`.

5.99.2.2 `template<typename _RAIter1, typename _RAIter2, typename _Pred > bool __gnu_parallel::__adjacent_find_selector`
`::operator() (_RAIter1 __i1, _RAIter2 __i2, _Pred __pred) \[inline\]`

Test on one position.

Parameters

<code>__i1</code>	_Iterator on first sequence.
<code>__i2</code>	_Iterator on second sequence (unused).
<code>__pred</code>	Find predicate.

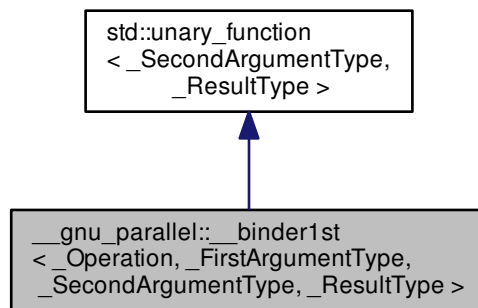
Definition at line 90 of file `find_selectors.h`.

The documentation for this struct was generated from the following file:

- [find_selectors.h](#)

5.100 `__gnu_parallel::__binder1st<_Operation, _FirstArgumentType, _SecondArgumentType, _ResultType >` Class Template Reference

Inheritance diagram for `__gnu_parallel::__binder1st<_Operation, _FirstArgumentType, _SecondArgumentType, _ResultType >`:



Public Types

- typedef `_SecondArgumentType` [argument_type](#)
- typedef `_ResultType` [result_type](#)

Public Member Functions

- **`__binder1st`** (`const _Operation &__x, const _FirstArgumentType &__y`)
- `_ResultType` **`operator()`** (`const _SecondArgumentType &__x`)
- `_ResultType` **`operator()`** (`_SecondArgumentType &__x`) `const`

Protected Attributes

- `_Operation` **`_M_op`**
- `_FirstArgumentType` **`_M_value`**

5.100.1 Detailed Description

```
template<typename _Operation, typename _FirstArgumentType, typename _SecondArgumentType, typename _ResultType>
class __gnu_parallel::__binder1st< _Operation, _FirstArgumentType, _SecondArgumentType, _ResultType >
```

Similar to `std::binder1st`, but giving the argument types explicitly.

Definition at line 192 of file `parallel/base.h`.

5.100.2 Member Typedef Documentation

5.100.2.1 `typedef _SecondArgumentType std::unary_function< _SecondArgumentType, _ResultType >::argument_type`
[*inherited*]

`argument_type` is the type of the argument

Definition at line 108 of file `stl_function.h`.

5.100.2.2 `typedef _ResultType std::unary_function< _SecondArgumentType, _ResultType >::result_type`
[*inherited*]

`result_type` is the return type

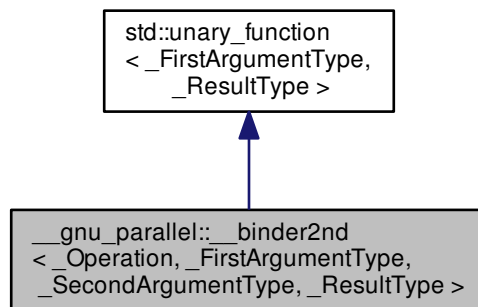
Definition at line 111 of file `stl_function.h`.

The documentation for this class was generated from the following file:

- [parallel/base.h](#)

5.101 `__gnu_parallel::__binder2nd< _Operation, _FirstArgumentType, _SecondArgumentType, _ResultType >` Class Template Reference

Inheritance diagram for `__gnu_parallel::__binder2nd< _Operation, _FirstArgumentType, _SecondArgumentType, _ResultType >`:



Public Types

- typedef `_FirstArgumentType` [argument_type](#)
- typedef `_ResultType` [result_type](#)

Public Member Functions

- `__binder2nd` (const `_Operation` &__x, const `_SecondArgumentType` &__y)
- `_ResultType operator()` (const `_FirstArgumentType` &__x) const
- `_ResultType operator()` (`_FirstArgumentType` &__x)

Protected Attributes

- `_Operation` **`_M_op`**
- `_SecondArgumentType` **`_M_value`**

5.101.1 Detailed Description

```
template<typename _Operation, typename _FirstArgumentType, typename _SecondArgumentType, typename _ResultType>
class __gnu_parallel::__binder2nd< _Operation, _FirstArgumentType, _SecondArgumentType, _ResultType >
```

Similar to `std::binder2nd`, but giving the argument types explicitly.

Definition at line 220 of file `parallel/base.h`.

5.101.2 Member Typedef Documentation

5.101.2.1 `typedef _FirstArgumentType std::unary_function< _FirstArgumentType , _ResultType >::argument_type` `[inherited]`

`argument_type` is the type of the argument

Definition at line 108 of file `stl_function.h`.

5.101.2.2 `typedef _ResultType std::unary_function< _FirstArgumentType , _ResultType >::result_type` `[inherited]`

`result_type` is the return type

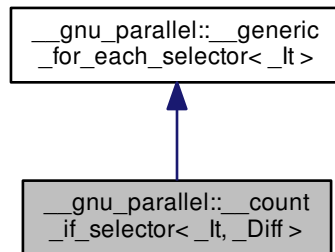
Definition at line 111 of file `stl_function.h`.

The documentation for this class was generated from the following file:

- [parallel/base.h](#)

5.102 `__gnu_parallel::__count_if_selector<_It, _Diff>` Struct Template Reference

Inheritance diagram for `__gnu_parallel::__count_if_selector<_It, _Diff>`:



Public Member Functions

- `template<typename _Op>`
`_Diff operator() (_Op &__o, _It __i)`

Public Attributes

- `_It _M_finish_iterator`

5.102.1 Detailed Description

```
template<typename _It, typename _Diff>
struct __gnu_parallel::__count_if_selector<_It, _Diff>
```

`std::count_if()` selector.

Definition at line 194 of file `for_each_selectors.h`.

5.102.2 Member Function Documentation

5.102.2.1 `template<typename _It, typename _Diff> template<typename _Op> _Diff __gnu_parallel::__count_if_selector<_It, _Diff>::operator() (_Op &__o, _It __i) [inline]`

Functor execution.

Parameters

<code>_↔ _o</code>	Operator.
<code>_↔ _i</code>	iterator referencing object.

Returns

1 if count, 0 if does not count.

Definition at line 202 of file `for_each_selectors.h`.

5.102.3 Member Data Documentation

5.102.3.1 `template<typename _It> _It __gnu_parallel::__generic_for_each_selector<_It>::M_finish_iterator`
[inherited]

`_Iterator` on last element processed; needed for some algorithms (e. g. `std::transform()`).

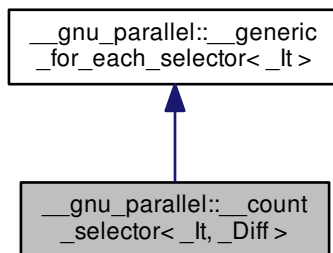
Definition at line 47 of file `for_each_selectors.h`.

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

5.103 __gnu_parallel::__count_selector<_It, _Diff> Struct Template Reference

Inheritance diagram for `__gnu_parallel::__count_selector<_It, _Diff>`:



Public Member Functions

- `template<typename _ValueType > _Diff operator() (_ValueType &__v, _It __i)`

Public Attributes

- `_It _M_finish_iterator`

5.103.1 Detailed Description

```
template<typename _It, typename _Diff>
struct __gnu_parallel::__count_selector<_It, _Diff>
```

`std::count()` selector.

Definition at line 180 of file `for_each_selectors.h`.

5.103.2 Member Function Documentation

5.103.2.1 `template<typename _It, typename _Diff> template<typename _ValueType > _Diff __gnu_parallel::__count_selector<_It, _Diff>::operator() (_ValueType & __v, _It __i)`
`[inline]`

Functor execution.

Parameters

<code>__v</code>	Current value.
<code>__i</code>	iterator referencing object.

Returns

1 if count, 0 if does not count.

Definition at line 188 of file `for_each_selectors.h`.

5.103.3 Member Data Documentation

5.103.3.1 `template<typename _It > _It __gnu_parallel::__generic_for_each_selector<_It>::_M_finish_iterator`
`[inherited]`

`_Iterator` on last element processed; needed for some algorithms (e. g. `std::transform()`).

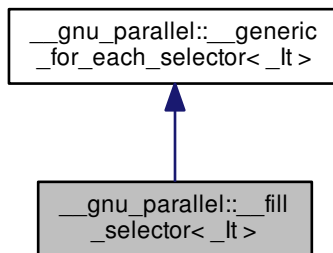
Definition at line 47 of file `for_each_selectors.h`.

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

5.104 `__gnu_parallel::__fill_selector<_It>` Struct Template Reference

Inheritance diagram for `__gnu_parallel::__fill_selector<_It>`:



Public Member Functions

- `template<typename _ValueType >`
`bool operator() (_ValueType &__v, _It __i)`

Public Attributes

- `_It _M_finish_iterator`

5.104.1 Detailed Description

```
template<typename _It>
struct __gnu_parallel::__fill_selector<_It>
```

`std::fill()` selector.

Definition at line 84 of file `for_each_selectors.h`.

5.104.2 Member Function Documentation

5.104.2.1 `template<typename _It > template<typename _ValueType > bool __gnu_parallel::__fill_selector<_It>::operator() (_ValueType &__v, _It __i) [inline]`

Functor execution.

Parameters

<code>__↔ _v</code>	Current value.
<code>__↔ _i</code>	iterator referencing object.

Definition at line 91 of file `for_each_selectors.h`.

5.104.3 Member Data Documentation

5.104.3.1 `template<typename _It> _It __gnu_parallel::__generic_for_each_selector<_It>::M_finish_iterator`
[inherited]

`_Iterator` on last element processed; needed for some algorithms (e. g. `std::transform()`).

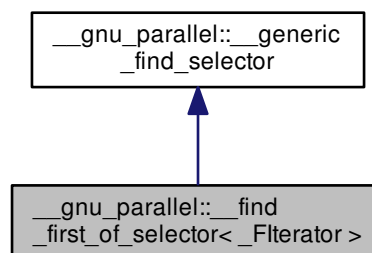
Definition at line 47 of file `for_each_selectors.h`.

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

5.105 `__gnu_parallel::__find_first_of_selector<_FIterator>` Struct Template Reference

Inheritance diagram for `__gnu_parallel::__find_first_of_selector<_FIterator>`:



Public Member Functions

- `__find_first_of_selector` (`_FIterator __begin`, `_FIterator __end`)
- `template<typename _RAIter1, typename _RAIter2, typename _Pred>`
`std::pair<_RAIter1, _RAIter2> _M_sequential_algorithm (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __↔
begin2, _Pred __pred)`
- `template<typename _RAIter1, typename _RAIter2, typename _Pred>`
`bool operator\(\) (_RAIter1 __i1, _RAIter2 __i2, _Pred __pred)`

Public Attributes

- `_Fiterator _M_begin`
- `_Fiterator _M_end`

5.105.1 Detailed Description

```
template<typename _Fiterator>
struct __gnu_parallel::__find_first_of_selector<_Fiterator >
```

Test predicate on several elements.

Definition at line 153 of file `find_selectors.h`.

5.105.2 Member Function Documentation

```
5.105.2.1 template<typename _Fiterator > template<typename _RAIter1 , typename _RAIter2 , typename
_Pred > std::pair<_RAIter1, _RAIter2> __gnu_parallel::__find_first_of_selector<_Fiterator
>::__M_sequential_algorithm ( _RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred ) [inline]
```

Corresponding sequential algorithm on a sequence.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence.
<code>__pred</code>	Find predicate.

Definition at line 186 of file `find_selectors.h`.

References `std::make_pair()`.

```
5.105.2.2 template<typename _Fiterator > template<typename _RAIter1 , typename _RAIter2 , typename _Pred > bool
__gnu_parallel::__find_first_of_selector<_Fiterator >::operator() ( _RAIter1 __i1, _RAIter2 __i2, _Pred __pred )
[inline]
```

Test on one position.

Parameters

<code>__i1</code>	_Iterator on first sequence.
<code>__i2</code>	_Iterator on second sequence (unused).
<code>__pred</code>	Find predicate.

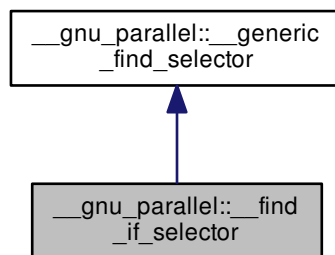
Definition at line 169 of file find_selectors.h.

The documentation for this struct was generated from the following file:

- [find_selectors.h](#)

5.106 __gnu_parallel::__find_if_selector Struct Reference

Inheritance diagram for __gnu_parallel::__find_if_selector:



Public Member Functions

- `template<typename _RAIter1, typename _RAIter2, typename _Pred >`
`std::pair<_RAIter1, _RAIter2 > M_sequential_algorithm (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred)`
- `template<typename _RAIter1, typename _RAIter2, typename _Pred >`
`bool operator\(\) (_RAIter1 __i1, _RAIter2 __i2, _Pred __pred)`

5.106.1 Detailed Description

Test predicate on a single element, used for `std::find()` and `std::find_if()`.

Definition at line 50 of file find_selectors.h.

5.106.2 Member Function Documentation

5.106.2.1 `template<typename _RAIter1, typename _RAIter2, typename _Pred > std::pair<_RAIter1, _RAIter2>`
`__gnu_parallel::__find_if_selector::M_sequential_algorithm (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred) [inline]`

Corresponding sequential algorithm on a sequence.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence.
<code>__pred</code>	Find predicate.

Definition at line 72 of file `find_selectors.h`.

References `std::make_pair()`.

5.106.2.2 `template<typename _RAIter1, typename _RAIter2, typename _Pred > bool __gnu_parallel::__find_if_selector::operator()
(_RAIter1 __i1, _RAIter2 __i2, _Pred __pred) [inline]`

Test on one position.

Parameters

<code>__i1</code>	_Iterator on first sequence.
<code>__i2</code>	_Iterator on second sequence (unused).
<code>__pred</code>	Find predicate.

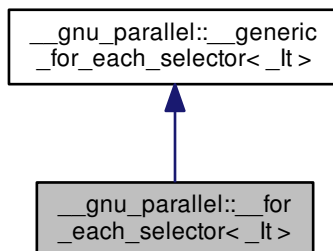
Definition at line 60 of file `find_selectors.h`.

The documentation for this struct was generated from the following file:

- [find_selectors.h](#)

5.107 `__gnu_parallel::__for_each_selector<_It >` Struct Template Reference

Inheritance diagram for `__gnu_parallel::__for_each_selector<_It >`:



Public Member Functions

- `template<typename _Op>`
`bool operator() (_Op &__o, _It __i)`

Public Attributes

- `_It _M_finish_iterator`

5.107.1 Detailed Description

```
template<typename _It>
struct __gnu_parallel::__for_each_selector<_It>
```

`std::for_each()` selector.

Definition at line 52 of file `for_each_selectors.h`.

5.107.2 Member Function Documentation

5.107.2.1 `template<typename _It> template<typename _Op> bool __gnu_parallel::__for_each_selector<_It>::operator() (_Op &__o, _It __i) [inline]`

Functor execution.

Parameters

<code>__o</code>	Operator.
<code>__i</code>	iterator referencing object.

Definition at line 59 of file `for_each_selectors.h`.

5.107.3 Member Data Documentation

5.107.3.1 `template<typename _It> _It __gnu_parallel::__generic_for_each_selector<_It>::_M_finish_iterator [inherited]`

`_Iterator` on last element processed; needed for some algorithms (e. g. `std::transform()`).

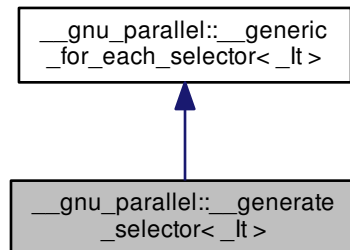
Definition at line 47 of file `for_each_selectors.h`.

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

5.108 `__gnu_parallel::__generate_selector<_It>` Struct Template Reference

Inheritance diagram for `__gnu_parallel::__generate_selector<_It>`:



Public Member Functions

- `template<typename _Op>`
`bool operator() (_Op &__o, _It __i)`

Public Attributes

- `_It __M_finish_iterator`

5.108.1 Detailed Description

```
template<typename _It>
struct __gnu_parallel::__generate_selector<_It>
```

`std::generate()` selector.

Definition at line 68 of file `for_each_selectors.h`.

5.108.2 Member Function Documentation

5.108.2.1 `template<typename _It> template<typename _Op> bool __gnu_parallel::__generate_selector<_It>::operator() (_Op &__o, _It __i) [inline]`

Functor execution.

Parameters

<code>_↔ _o</code>	Operator.
<code>_↔ _i</code>	iterator referencing object.

Definition at line 75 of file `for_each_selectors.h`.

5.108.3 Member Data Documentation

5.108.3.1 `template<typename _It> _It __gnu_parallel::__generic_for_each_selector<_It>::_M_finish_iterator`
[inherited]

`_Iterator` on last element processed; needed for some algorithms (e. g. `std::transform()`).

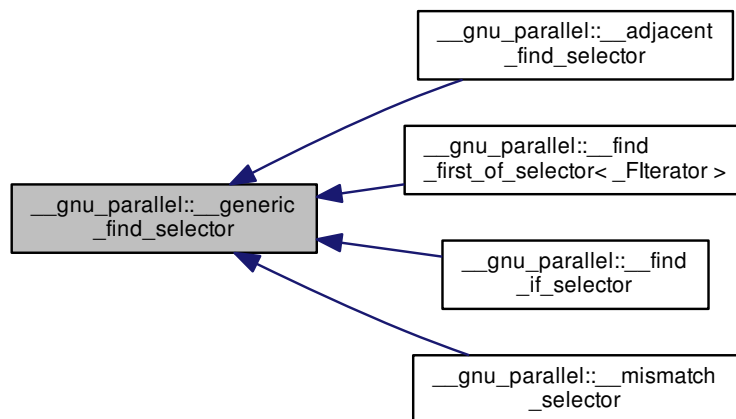
Definition at line 47 of file `for_each_selectors.h`.

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

5.109 `__gnu_parallel::__generic_find_selector` Struct Reference

Inheritance diagram for `__gnu_parallel::__generic_find_selector`:



5.109.1 Detailed Description

Base class of all `__gnu_parallel::__find_template` selectors.

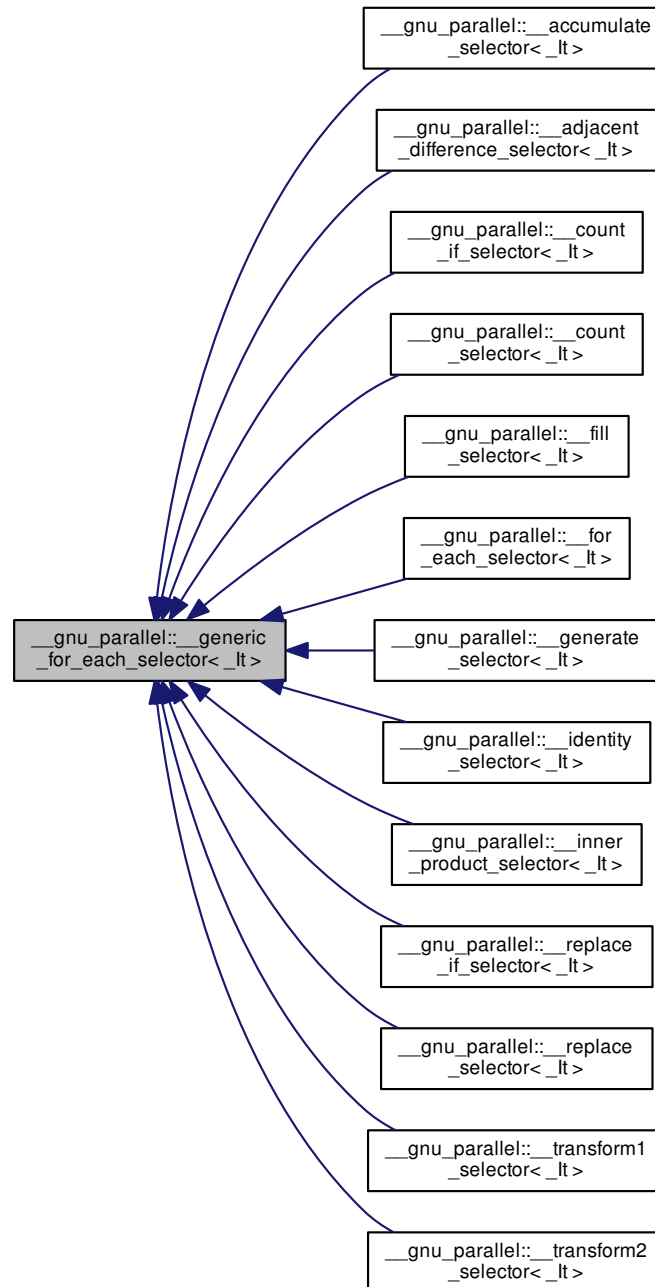
Definition at line 43 of file `find_selectors.h`.

The documentation for this struct was generated from the following file:

- [find_selectors.h](#)

5.110 __gnu_parallel::__generic_for_each_selector<_It> Struct Template Reference

Inheritance diagram for __gnu_parallel::__generic_for_each_selector<_It>:



Public Attributes

- [_It _M_finish_iterator](#)

5.110.1 Detailed Description

```
template<typename _It>
struct __gnu_parallel::__generic_for_each_selector< _It >
```

Generic __selector for embarrassingly parallel functions.

Definition at line 42 of file `for_each_selectors.h`.

5.110.2 Member Data Documentation

5.110.2.1 `template<typename _It > _It __gnu_parallel::__generic_for_each_selector< _It >::_M_finish_iterator`

`_Iterator` on last element processed; needed for some algorithms (e. g. `std::transform()`).

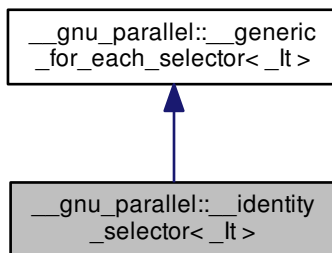
Definition at line 47 of file `for_each_selectors.h`.

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

5.111 `__gnu_parallel::__identity_selector< _It >` Struct Template Reference

Inheritance diagram for `__gnu_parallel::__identity_selector< _It >`:



Public Member Functions

- `template<typename _Op>
_It operator() (_Op __o, _It __i)`

Public Attributes

- `_It _M_finish_iterator`

5.111.1 Detailed Description

```
template<typename _It>
struct __gnu_parallel::__identity_selector<_It>
```

Selector that just returns the passed iterator.

Definition at line 253 of file `for_each_selectors.h`.

5.111.2 Member Function Documentation

5.111.2.1 `template<typename _It> template<typename _Op> _It __gnu_parallel::__identity_selector<_It>::operator() (_Op __o, _It __i) [inline]`

Functor execution.

Parameters

<code>__o</code>	Operator (unused).
<code>__i</code>	iterator referencing object.

Returns

Passed iterator.

Definition at line 261 of file `for_each_selectors.h`.

5.111.3 Member Data Documentation

5.111.3.1 `template<typename _It> _It __gnu_parallel::__generic_for_each_selector<_It>::_M_finish_iterator [inherited]`

`_Iterator` on last element processed; needed for some algorithms (e. g. `std::transform()`).

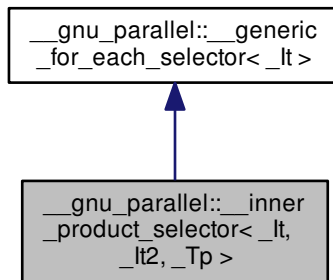
Definition at line 47 of file `for_each_selectors.h`.

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

5.112 `__gnu_parallel::__inner_product_selector<_It, _It2, _Tp>` Struct Template Reference

Inheritance diagram for `__gnu_parallel::__inner_product_selector<_It, _It2, _Tp>`:



Public Member Functions

- [__inner_product_selector](#) (`_It __b1, _It2 __b2`)
- `template<typename _Op>`
`_Tp operator()` (`_Op __mult, _It __current`)

Public Attributes

- `_It __begin1_iterator`
- `_It2 __begin2_iterator`
- `_It __M_finish_iterator`

5.112.1 Detailed Description

```
template<typename _It, typename _It2, typename _Tp>
struct __gnu_parallel::__inner_product_selector<_It, _It2, _Tp>
```

`std::inner_product()` selector.

Definition at line 222 of file `for_each_selectors.h`.

5.112.2 Constructor & Destructor Documentation

5.112.2.1 `template<typename _It, typename _It2, typename _Tp> __gnu_parallel::__inner_product_selector<_It, _It2, _Tp>::__inner_product_selector (_It __b1, _It2 __b2)` `[inline], [explicit]`

Constructor.

Parameters

<code>__b1</code>	Begin iterator of first sequence.
<code>__b2</code>	Begin iterator of second sequence.

Definition at line 234 of file `for_each_selectors.h`.

5.112.3 Member Function Documentation

5.112.3.1 `template<typename _It, typename _It2, typename _Tp> template<typename _Op> _Tp
__gnu_parallel::__inner_product_selector<_It,_It2,_Tp>::operator() (_Op __mult, _It __current)
[inline]`

Functor execution.

Parameters

<code>__mult</code>	Multiplication functor.
<code>__current</code>	iterator referencing object.

Returns

Inner product elemental `__result`.

Definition at line 243 of file `for_each_selectors.h`.

5.112.4 Member Data Documentation

5.112.4.1 `template<typename _It, typename _It2, typename _Tp> _It __gnu_parallel::__inner_product_selector<_It,_It2,_Tp>::__begin1_iterator`

Begin iterator of first sequence.

Definition at line 225 of file `for_each_selectors.h`.

5.112.4.2 `template<typename _It, typename _It2, typename _Tp> _It2 __gnu_parallel::__inner_product_selector<_It,_It2,_Tp>::__begin2_iterator`

Begin iterator of second sequence.

Definition at line 228 of file `for_each_selectors.h`.

5.112.4.3 `template<typename _It > _It __gnu_parallel::__generic_for_each_selector< _It >::M_finish_iterator` [inherited]

`_Iterator` on last element processed; needed for some algorithms (e. g. `std::transform()`).

Definition at line 47 of file `for_each_selectors.h`.

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

5.113 `__gnu_parallel::__max_element_reduct< _Compare, _It >` Struct Template Reference

Public Member Functions

- `__max_element_reduct` (`_Compare &__c`)
- `_It operator()` (`_It __x, _It __y`)

Public Attributes

- `_Compare & __comp`

5.113.1 Detailed Description

```
template<typename _Compare, typename _It>
struct __gnu_parallel::__max_element_reduct< _Compare, _It >
```

Reduction for finding the maximum element, using a comparator.

Definition at line 321 of file `for_each_selectors.h`.

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

5.114 `__gnu_parallel::__min_element_reduct< _Compare, _It >` Struct Template Reference

Public Member Functions

- `__min_element_reduct` (`_Compare &__c`)
- `_It operator()` (`_It __x, _It __y`)

Public Attributes

- `_Compare` & `__comp`

5.114.1 Detailed Description

```
template<typename _Compare, typename _It>
struct __gnu_parallel::__min_element_reduct< _Compare, _It >
```

Reduction for finding the maximum element, using a comparator.

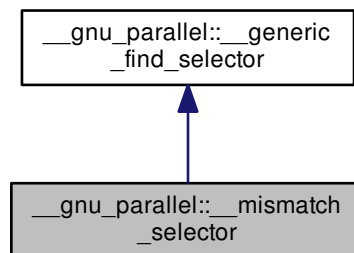
Definition at line 307 of file `for_each_selectors.h`.

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

5.115 `__gnu_parallel::__mismatch_selector` Struct Reference

Inheritance diagram for `__gnu_parallel::__mismatch_selector`:



Public Member Functions

- `template<typename _RAIter1, typename _RAIter2, typename _Pred >`
`std::pair<_RAIter1, _RAIter2 > _M_sequential_algorithm (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred)`
- `template<typename _RAIter1, typename _RAIter2, typename _Pred >`
`bool operator\(\) (_RAIter1 __i1, _RAIter2 __i2, _Pred __pred)`

5.115.1 Detailed Description

Test inverted predicate on a single element.

Definition at line 119 of file `find_selectors.h`.

5.115.2 Member Function Documentation

5.115.2.1 `template<typename _RAIter1, typename _RAIter2, typename _Pred> std::pair<_RAIter1, _RAIter2> __gnu_parallel::__mismatch_selector::M_sequential_algorithm (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred) [inline]`

Corresponding sequential algorithm on a sequence.

Parameters

<code>__begin1</code>	Begin iterator of first sequence.
<code>__end1</code>	End iterator of first sequence.
<code>__begin2</code>	Begin iterator of second sequence.
<code>__pred</code>	Find predicate.

Definition at line 143 of file `find_selectors.h`.

5.115.2.2 `template<typename _RAIter1, typename _RAIter2, typename _Pred> bool __gnu_parallel::__mismatch_selector::operator() (_RAIter1 __i1, _RAIter2 __i2, _Pred __pred) [inline]`

Test on one position.

Parameters

<code>__i1</code>	_Iterator on first sequence.
<code>__i2</code>	_Iterator on second sequence (unused).
<code>__pred</code>	Find predicate.

Definition at line 130 of file `find_selectors.h`.

The documentation for this struct was generated from the following file:

- [find_selectors.h](#)

5.116 `__gnu_parallel::__multiway_merge_3_variant_sentinel_switch< __sentinels, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare>` Struct Template Reference

Public Member Functions

- `_RAIter3 operator() (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 __target, _DifferenceTp __length, _Compare __comp)`

5.116.1 Detailed Description

```
template<bool __sentinels, typename _RAIterlterator, typename _RAIter3, typename _DifferenceTp, typename _Compare>
struct __gnu_parallel::__multiway_merge_3_variant_sentinel_switch< __sentinels, _RAIterlterator, _RAIter3, _DifferenceTp, _Compare
>
```

Switch for 3-way merging with `__sentinels` turned off.

Note that 3-way merging is always stable!

Definition at line 752 of file `multiway_merge.h`.

The documentation for this struct was generated from the following file:

- [multiway_merge.h](#)

5.117 `__gnu_parallel::__multiway_merge_3_variant_sentinel_switch< true, _RAIterlterator, _RAIter3, _DifferenceTp, _Compare > Struct Template Reference`

Public Member Functions

- `_RAIter3 operator() (_RAIterlterator __seqs_begin, _RAIterlterator __seqs_end, _RAIter3 __target, _DifferenceTp __length, _Compare __comp)`

5.117.1 Detailed Description

```
template<typename _RAIterlterator, typename _RAIter3, typename _DifferenceTp, typename _Compare>
struct __gnu_parallel::__multiway_merge_3_variant_sentinel_switch< true, _RAIterlterator, _RAIter3, _DifferenceTp, _Compare >
```

Switch for 3-way merging with `__sentinels` turned on.

Note that 3-way merging is always stable!

Definition at line 772 of file `multiway_merge.h`.

The documentation for this struct was generated from the following file:

- [multiway_merge.h](#)

5.118 `__gnu_parallel::__multiway_merge_4_variant_sentinel_switch< __sentinels, _RAIterlterator, _RAIter3, _DifferenceTp, _Compare > Struct Template Reference`

Public Member Functions

- `_RAIter3 operator() (_RAIterlterator __seqs_begin, _RAIterlterator __seqs_end, _RAIter3 __target, _DifferenceTp __length, _Compare __comp)`

5.118.1 Detailed Description

```
template<bool __sentinels, typename _RAIterIterator, typename _RAIter3, typename _DifferenceTp, typename _Compare>
struct __gnu_parallel::__multiway_merge_4_variant_sentinel_switch< __sentinels, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare
>
```

Switch for 4-way merging with `__sentinels` turned off.

Note that 4-way merging is always stable!

Definition at line 795 of file `multiway_merge.h`.

The documentation for this struct was generated from the following file:

- [multiway_merge.h](#)

5.119 `__gnu_parallel::__multiway_merge_4_variant_sentinel_switch< true, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >` Struct Template Reference

Public Member Functions

- `_RAIter3 operator() (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 __target, _DifferenceTp __length, _Compare __comp)`

5.119.1 Detailed Description

```
template<typename _RAIterIterator, typename _RAIter3, typename _DifferenceTp, typename _Compare>
struct __gnu_parallel::__multiway_merge_4_variant_sentinel_switch< true, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >
```

Switch for 4-way merging with `__sentinels` turned on.

Note that 4-way merging is always stable!

Definition at line 815 of file `multiway_merge.h`.

The documentation for this struct was generated from the following file:

- [multiway_merge.h](#)

5.120 `__gnu_parallel::__multiway_merge_k_variant_sentinel_switch< __sentinels, __stable, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >` Struct Template Reference

Public Member Functions

- `_RAIter3 operator() (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 __target, const typename std::iterator_traits< typename std::iterator_traits< _RAIterIterator >::value_type::first_type >::value_type & __sentinel, _DifferenceTp __length, _Compare __comp)`

5.120.1 Detailed Description

```
template<bool __sentinels, bool __stable, typename _RAIterlterator, typename _RAIter3, typename _DifferenceTp, typename _Compare>
struct __gnu_parallel::__multiway_merge_k_variant_sentinel_switch< __sentinels, __stable, _RAIterlterator, _RAIter3, _DifferenceTp, _Compare >
```

Switch for k-way merging with `__sentinels` turned on.

Definition at line 837 of file `multiway_merge.h`.

The documentation for this struct was generated from the following file:

- [multiway_merge.h](#)

5.121 `__gnu_parallel::__multiway_merge_k_variant_sentinel_switch< false, __stable, _RAIterlterator, _RAIter3, _DifferenceTp, _Compare >` Struct Template Reference

Public Member Functions

- `_RAIter3 operator() (_RAIterlterator __seqs_begin, _RAIterlterator __seqs_end, _RAIter3 __target, const typename std::iterator_traits< typename std::iterator_traits< _RAIterlterator >::value_type::first_type >::value_type & __sentinel, _DifferenceTp __length, _Compare __comp)`

5.121.1 Detailed Description

```
template<bool __stable, typename _RAIterlterator, typename _RAIter3, typename _DifferenceTp, typename _Compare>
struct __gnu_parallel::__multiway_merge_k_variant_sentinel_switch< false, __stable, _RAIterlterator, _RAIter3, _DifferenceTp, _Compare >
```

Switch for k-way merging with `__sentinels` turned off.

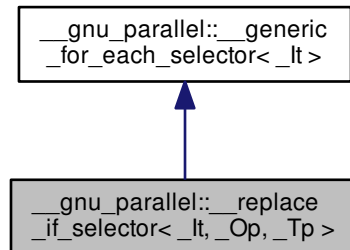
Definition at line 872 of file `multiway_merge.h`.

The documentation for this struct was generated from the following file:

- [multiway_merge.h](#)

5.122 `__gnu_parallel::__replace_if_selector<_It,_Op,_Tp>` Struct Template Reference

Inheritance diagram for `__gnu_parallel::__replace_if_selector<_It,_Op,_Tp>`:



Public Member Functions

- `__replace_if_selector` (const `_Tp` & `__new_val`)
- bool `operator()` (`_Op` & `__o`, `_It` `__i`)

Public Attributes

- const `_Tp` & `__new_val`
- `_It` `_M_finish_iterator`

5.122.1 Detailed Description

```
template<typename _It, typename _Op, typename _Tp>
struct __gnu_parallel::__replace_if_selector<_It,_Op,_Tp>
```

`std::replace()` selector.

Definition at line 156 of file `for_each_selectors.h`.

5.122.2 Constructor & Destructor Documentation

5.122.2.1 `template<typename _It, typename _Op, typename _Tp> __gnu_parallel::__replace_if_selector<_It,_Op,_Tp>::__replace_if_selector (const _Tp & __new_val)` `[inline]`, `[explicit]`

Constructor.

Parameters

<code>__new_val</code>	Value to replace with.
------------------------	------------------------

Definition at line 164 of file `for_each_selectors.h`.

5.122.3 Member Function Documentation

5.122.3.1 `template<typename _It, typename _Op, typename _Tp> bool __gnu_parallel::__replace_if_selector<_It,_Op,_Tp>::operator()(_Op &__o, _It __i) [inline]`

Functor execution.

Parameters

<code>__o</code>	Operator.
<code>__i</code>	iterator referencing object.

Definition at line 170 of file `for_each_selectors.h`.

5.122.4 Member Data Documentation

5.122.4.1 `template<typename _It, typename _Op, typename _Tp> const _Tp& __gnu_parallel::__replace_if_selector<_It,_Op,_Tp>::__new_val`

Value to replace with.

Definition at line 159 of file `for_each_selectors.h`.

5.122.4.2 `template<typename _It> _It __gnu_parallel::__generic_for_each_selector<_It>::M_finish_iterator [inherited]`

`_Iterator` on last element processed; needed for some algorithms (e. g. `std::transform()`).

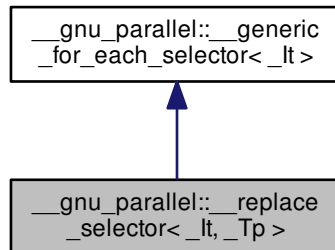
Definition at line 47 of file `for_each_selectors.h`.

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

5.123 `__gnu_parallel::__replace_selector<_It, _Tp>` Struct Template Reference

Inheritance diagram for `__gnu_parallel::__replace_selector<_It, _Tp>`:



Public Member Functions

- `__replace_selector` (const `_Tp` & `__new_val`)
- bool `operator()` (`_Tp` & `__v`, `_It` `__i`)

Public Attributes

- const `_Tp` & `__new_val`
- `_It` `__M_finish_iterator`

5.123.1 Detailed Description

```
template<typename _It, typename _Tp>
struct __gnu_parallel::__replace_selector<_It, _Tp>
```

`std::replace()` selector.

Definition at line 132 of file `for_each_selectors.h`.

5.123.2 Constructor & Destructor Documentation

5.123.2.1 `template<typename _It, typename _Tp> __gnu_parallel::__replace_selector<_It, _Tp>::__replace_selector (const _Tp & __new_val) [inline], [explicit]`

Constructor.

Parameters

<code>__new_val</code>	Value to replace with.
------------------------	------------------------

Definition at line 140 of file `for_each_selectors.h`.

5.123.3 Member Function Documentation

5.123.3.1 `template<typename _It, typename _Tp> bool __gnu_parallel::__replace_selector<_It,_Tp>::operator() (_Tp & __v, _It __i) [inline]`

Functor execution.

Parameters

<code>__v</code>	Current value.
<code>__i</code>	iterator referencing object.

Definition at line 146 of file `for_each_selectors.h`.

5.123.4 Member Data Documentation

5.123.4.1 `template<typename _It, typename _Tp> const _Tp& __gnu_parallel::__replace_selector<_It,_Tp>::__new_val`

Value to replace with.

Definition at line 135 of file `for_each_selectors.h`.

5.123.4.2 `template<typename _It> _It __gnu_parallel::__generic_for_each_selector<_It>::__M_finish_iterator [inherited]`

`_It` iterator on last element processed; needed for some algorithms (e. g. `std::transform()`).

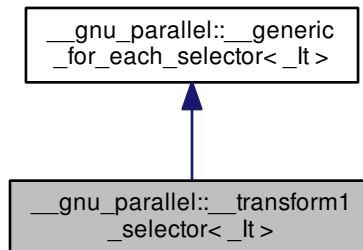
Definition at line 47 of file `for_each_selectors.h`.

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

5.124 `__gnu_parallel::__transform1_selector<_It>` Struct Template Reference

Inheritance diagram for `__gnu_parallel::__transform1_selector<_It>`:



Public Member Functions

- `template<typename _Op>`
`bool operator() (_Op &__o, _It __i)`

Public Attributes

- `_It __M_finish_iterator`

5.124.1 Detailed Description

```
template<typename _It>
struct __gnu_parallel::__transform1_selector<_It>
```

`std::transform()` __selector, one input sequence variant.

Definition at line 100 of file `for_each_selectors.h`.

5.124.2 Member Function Documentation

5.124.2.1 `template<typename _It> template<typename _Op> bool __gnu_parallel::__transform1_selector<_It>::operator() (_Op &__o, _It __i) [inline]`

Functor execution.

Parameters

<code>__o</code>	Operator.
<code>__i</code>	iterator referencing object.

Definition at line 107 of file `for_each_selectors.h`.

5.124.3 Member Data Documentation

5.124.3.1 `template<typename _It> _It __gnu_parallel::__generic_for_each_selector<_It>::M_finish_iterator`
[inherited]

`_Iterator` on last element processed; needed for some algorithms (e. g. `std::transform()`).

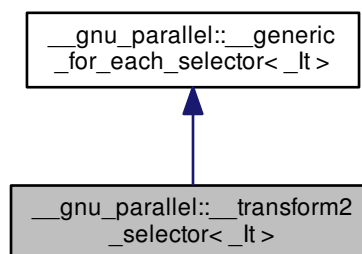
Definition at line 47 of file `for_each_selectors.h`.

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

5.125 `__gnu_parallel::__transform2_selector<_It>` Struct Template Reference

Inheritance diagram for `__gnu_parallel::__transform2_selector<_It>`:



Public Member Functions

- `template<typename _Op>`
`bool operator() (_Op &__o, _It __i)`

Public Attributes

- [_lt _M_finish_iterator](#)

5.125.1 Detailed Description

```
template<typename _It>
struct __gnu_parallel::__transform2_selector< _It >
```

std::transform() __selector, two input sequences variant.



Definition at line 116 of file for_each_selectors.h.

5.125.2 Member Function Documentation

5.125.2.1 `template<typename _It> template<typename _Op > bool __gnu_parallel::__transform2_selector< _It >::operator()(_Op &__o, _It __i) [inline]`

Functor execution.

Parameters

 <code>__o</code>	Operator.
 <code>__i</code>	iterator referencing object.

Definition at line 123 of file for_each_selectors.h.

5.125.3 Member Data Documentation

5.125.3.1 `template<typename _It > _lt __gnu_parallel::__generic_for_each_selector< _It >::__M_finish_iterator [inherited]`

_Iterator on last element processed; needed for some algorithms (e. g. std::transform()).

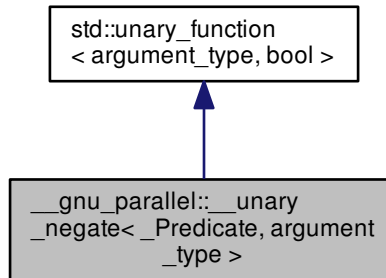
Definition at line 47 of file for_each_selectors.h.

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

5.126 `__gnu_parallel::__unary_negate<_Predicate, argument_type>` Class Template Reference

Inheritance diagram for `__gnu_parallel::__unary_negate<_Predicate, argument_type>`:



Public Types

- typedef `argument_type` `argument_type`
- typedef `bool` `result_type`

Public Member Functions

- `__unary_negate` (const `_Predicate` &__x)
- `bool operator()` (const `argument_type` &__x)

Protected Attributes

- `_Predicate` `_M_pred`

5.126.1 Detailed Description

```
template<typename _Predicate, typename argument_type>
class __gnu_parallel::__unary_negate<_Predicate, argument_type>
```

Similar to `std::unary_negate`, but giving the argument types explicitly.

Definition at line 173 of file `parallel/base.h`.

5.126.2 Member Typedef Documentation

5.126.2.1 `typedef argument_type std::unary_function< argument_type, bool >::argument_type` [inherited]

`argument_type` is the type of the argument

Definition at line 108 of file `stl_function.h`.

5.126.2.2 `typedef bool std::unary_function< argument_type, bool >::result_type` [inherited]

`result_type` is the return type

Definition at line 111 of file `stl_function.h`.

The documentation for this class was generated from the following file:

- [parallel/base.h](#)

5.127 `__gnu_parallel::_DRandomShufflingGlobalData< _RAIter >` Struct Template Reference

Public Types

- `typedef _TraitsType::difference_type` **DifferenceType**
- `typedef std::iterator_traits< _RAIter >` **TraitsType**
- `typedef _TraitsType::value_type` **ValueType**

Public Member Functions

- [_DRandomShufflingGlobalData](#) (`_RAIter &__source`)

Public Attributes

- [_ThreadIndex](#) * [_M_bin_proc](#)
- `DifferenceType` ** [_M_dist](#)
- `int` [_M_num_bins](#)
- `int` [_M_num_bits](#)
- `_RAIter &` [_M_source](#)
- `DifferenceType` * [_M_starts](#)
- `ValueType` ** [_M_temporaries](#)

5.127.1 Detailed Description

```
template<typename _RAIter>
struct __gnu_parallel::_DRandomShufflingGlobalData< _RAIter >
```

Data known to every thread participating in `__gnu_parallel::__parallel_random_shuffle()`.

Definition at line 52 of file `random_shuffle.h`.

5.127.2 Constructor & Destructor Documentation

5.127.2.1 `template<typename _RAIter> __gnu_parallel::_DRandomShufflingGlobalData<_RAIter>::_DRandomShufflingGlobalData(_RAIter & __source) [inline]`

Constructor.

Definition at line 83 of file `random_shuffle.h`.

5.127.3 Member Data Documentation

5.127.3.1 `template<typename _RAIter> _ThreadIndex* __gnu_parallel::_DRandomShufflingGlobalData<_RAIter>::_M_bin_proc`

Number of the thread that will further process the corresponding bin.

Definition at line 74 of file `random_shuffle.h`.

Referenced by `__gnu_parallel::__parallel_random_shuffle_drs()`.

5.127.3.2 `template<typename _RAIter> _DifferenceType** __gnu_parallel::_DRandomShufflingGlobalData<_RAIter>::_M_dist`

Two-dimensional array to hold the thread-bin distribution.

Dimensions `(_M_num_threads + 1) __x (_M_num_bins + 1)`.

Definition at line 67 of file `random_shuffle.h`.

Referenced by `__gnu_parallel::__parallel_random_shuffle_drs()`, and `__gnu_parallel::__parallel_random_shuffle_drs↵_pu()`.

5.127.3.3 `template<typename _RAIter> int __gnu_parallel::_DRandomShufflingGlobalData<_RAIter>::_M_num_bins`

Number of bins to distribute to.

Definition at line 77 of file `random_shuffle.h`.

Referenced by `__gnu_parallel::__parallel_random_shuffle_drs()`, and `__gnu_parallel::__parallel_random_shuffle_drs↵_pu()`.

5.127.3.4 `template<typename _RAIter> int __gnu_parallel::_DRandomShufflingGlobalData<_RAIter>::_M_num_bits`

Number of bits needed to address the bins.

Definition at line 80 of file `random_shuffle.h`.

Referenced by `__gnu_parallel::__parallel_random_shuffle_drs()`, and `__gnu_parallel::__parallel_random_shuffle_drs↵_pu()`.

5.127.3.5 `template<typename _RAIter> _RAIter& __gnu_parallel::_DRandomShufflingGlobalData< _RAIter>::__M_source`

Begin iterator of the `__source`.

Definition at line 59 of file `random_shuffle.h`.

5.127.3.6 `template<typename _RAIter> _DifferenceType* __gnu_parallel::_DRandomShufflingGlobalData< _RAIter>::__M_starts`

Start indexes of the threads' `__chunks`.

Definition at line 70 of file `random_shuffle.h`.

Referenced by `__gnu_parallel::__parallel_random_shuffle_drs()`, and `__gnu_parallel::__parallel_random_shuffle_drs↔_pu()`.

5.127.3.7 `template<typename _RAIter> _ValueType** __gnu_parallel::_DRandomShufflingGlobalData< _RAIter>::__M_temporaries`

Temporary arrays for each thread.

Definition at line 62 of file `random_shuffle.h`.

Referenced by `__gnu_parallel::__parallel_random_shuffle_drs()`.

The documentation for this struct was generated from the following file:

- [random_shuffle.h](#)

5.128 `__gnu_parallel::_DRSSorterPU< _RAIter, _RandomNumberGenerator >` Struct Template Reference

Public Attributes

- [_BinIndex __bins_end](#)
- [_BinIndex _M_bins_begin](#)
- [int _M_num_threads](#)
- [_DRandomShufflingGlobalData< _RAIter > * _M_sd](#)
- [uint32_t _M_seed](#)

5.128.1 Detailed Description

```
template<typename _RAIter, typename _RandomNumberGenerator>
struct __gnu_parallel::_DRSSorterPU< _RAIter, _RandomNumberGenerator >
```

Local data for a thread participating in `__gnu_parallel::__parallel_random_shuffle()`.

Definition at line 91 of file `random_shuffle.h`.

5.128.2 Member Data Documentation

5.128.2.1 `template<typename _RAIter, typename _RandomNumberGenerator> _BinIndex __gnu_parallel::_DRSSorterPU<_RAIter, _RandomNumberGenerator>::_bins_end`

End index for bins taken care of by this thread.

Definition at line 100 of file `random_shuffle.h`.

Referenced by `__gnu_parallel::_parallel_random_shuffle_drs()`.

5.128.2.2 `template<typename _RAIter, typename _RandomNumberGenerator> _BinIndex __gnu_parallel::_DRSSorterPU<_RAIter, _RandomNumberGenerator>::_M_bins_begin`

Begin index for bins taken care of by this thread.

Definition at line 97 of file `random_shuffle.h`.

Referenced by `__gnu_parallel::_parallel_random_shuffle_drs()`.

5.128.2.3 `template<typename _RAIter, typename _RandomNumberGenerator> int __gnu_parallel::_DRSSorterPU<_RAIter, _RandomNumberGenerator>::_M_num_threads`

Number of threads participating in total.

Definition at line 94 of file `random_shuffle.h`.

Referenced by `__gnu_parallel::_parallel_random_shuffle_drs()`, and `__gnu_parallel::_parallel_random_shuffle_drs←_pu()`.

5.128.2.4 `template<typename _RAIter, typename _RandomNumberGenerator> _DRandomShufflingGlobalData<_RAIter>* __gnu_parallel::_DRSSorterPU<_RAIter, _RandomNumberGenerator>::_M_sd`

Pointer to global data.

Definition at line 106 of file `random_shuffle.h`.

Referenced by `__gnu_parallel::_parallel_random_shuffle_drs()`, and `__gnu_parallel::_parallel_random_shuffle_drs←_pu()`.

5.128.2.5 `template<typename _RAIter, typename _RandomNumberGenerator> uint32_t __gnu_parallel::_DRSSorterPU<_RAIter, _RandomNumberGenerator>::_M_seed`

Random `_M_seed` for this thread.

Definition at line 103 of file `random_shuffle.h`.

Referenced by `__gnu_parallel::_parallel_random_shuffle_drs()`, and `__gnu_parallel::_parallel_random_shuffle_drs←_pu()`.

The documentation for this struct was generated from the following file:

- [random_shuffle.h](#)

5.129 `__gnu_parallel::_DummyReduct` Struct Reference

Public Member Functions

- `bool operator() (bool, bool) const`

5.129.1 Detailed Description

Reduction function doing nothing.

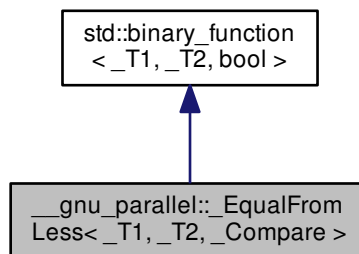
Definition at line 298 of file `for_each_selectors.h`.

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

5.130 `__gnu_parallel::_EqualFromLess<_T1, _T2, _Compare>` Class Template Reference

Inheritance diagram for `__gnu_parallel::_EqualFromLess<_T1, _T2, _Compare>`:



Public Types

- `typedef _T1 first_argument_type`
- `typedef bool result_type`
- `typedef _T2 second_argument_type`

Public Member Functions

- `_EqualFromLess (_Compare &__comp)`
- `bool operator() (const _T1 &__a, const _T2 &__b)`

5.130.1 Detailed Description

```
template<typename _T1, typename _T2, typename _Compare>
class __gnu_parallel::_EqualFromLess<_T1, _T2, _Compare>
```

Constructs predicate for equality from strict weak ordering predicate.

Definition at line 157 of file `parallel/base.h`.

5.130.2 Member Typedef Documentation

5.130.2.1 `typedef _T1 std::binary_function<_T1, _T2, bool>::first_argument_type` `[inherited]`

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.130.2.2 `typedef bool std::binary_function<_T1, _T2, bool>::result_type` `[inherited]`

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.130.2.3 `typedef _T2 std::binary_function<_T1, _T2, bool>::second_argument_type` `[inherited]`

`second_argument_type` is the type of the second argument

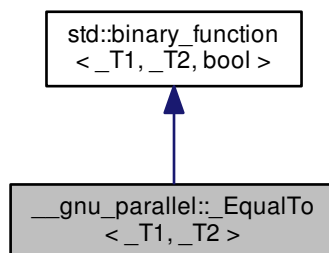
Definition at line 124 of file `stl_function.h`.

The documentation for this class was generated from the following file:

- [parallel/base.h](#)

5.131 `__gnu_parallel::_EqualTo<_T1, _T2>` Struct Template Reference

Inheritance diagram for `__gnu_parallel::_EqualTo<_T1, _T2>`:



Public Types

- typedef `_T1` [first_argument_type](#)
- typedef `bool` [result_type](#)
- typedef `_T2` [second_argument_type](#)

Public Member Functions

- `bool` **operator()** (`const _T1 &__t1, const _T2 &__t2`) `const`

5.131.1 Detailed Description

```
template<typename _T1, typename _T2>
struct __gnu_parallel::_EqualTo<_T1, _T2 >
```

Similar to `std::equal_to`, but allows two different types.

Definition at line 244 of file `parallel/base.h`.

5.131.2 Member Typedef Documentation

5.131.2.1 `typedef _T1 std::binary_function<_T1, _T2, bool >::first_argument_type` `[inherited]`

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.131.2.2 `typedef bool std::binary_function<_T1, _T2, bool >::result_type` `[inherited]`

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.131.2.3 `typedef _T2 std::binary_function<_T1, _T2, bool >::second_argument_type` `[inherited]`

`second_argument_type` is the type of the second argument

Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [parallel/base.h](#)

5.132 `__gnu_parallel::_GuardedIterator<_RAIter, _Compare>` Class Template Reference

Public Member Functions

- `_GuardedIterator` (`_RAIter __begin`, `_RAIter __end`, `_Compare &__comp`)
- `operator _RAIter` ()
- `std::iterator_traits<_RAIter>::value_type & operator*` ()
- `_GuardedIterator<_RAIter, _Compare> & operator++` ()

Friends

- `bool operator<` (`_GuardedIterator<_RAIter, _Compare> &__bi1`, `_GuardedIterator<_RAIter, _Compare> &__bi2`)
- `bool operator<=` (`_GuardedIterator<_RAIter, _Compare> &__bi1`, `_GuardedIterator<_RAIter, _Compare> &__bi2`)

5.132.1 Detailed Description

```
template<typename _RAIter, typename _Compare>
class __gnu_parallel::_GuardedIterator<_RAIter, _Compare>
```

`_Iterator` wrapper supporting an implicit supremum at the end of the sequence, dominating all comparisons.

The implicit supremum comes with a performance cost.

Deriving from `_RAIter` is not possible since `_RAIter` need not be a class.

Definition at line 73 of file `multiway_merge.h`.

5.132.2 Constructor & Destructor Documentation

5.132.2.1 `template<typename _RAIter, typename _Compare> __gnu_parallel::_GuardedIterator<_RAIter, _Compare>::_GuardedIterator (_RAIter __begin, _RAIter __end, _Compare &__comp)` `[inline]`

Constructor. Sets iterator to beginning of sequence.

Parameters

<code>__begin</code>	Begin iterator of sequence.
<code>__end</code>	End iterator of sequence.
<code>__comp</code>	Comparator provided for associated overloaded compare operators.

Definition at line 91 of file `multiway_merge.h`.

5.132.3 Member Function Documentation

5.132.3.1 `template<typename _RAIter, typename _Compare > __gnu_parallel::_GuardedIterator< _RAIter, _Compare >::operator _RAIter () [inline]`

Convert to wrapped iterator.

Returns

Wrapped iterator.

Definition at line 112 of file multiway_merge.h.

5.132.3.2 `template<typename _RAIter, typename _Compare > std::iterator_traits< _RAIter >::value_type& __gnu_parallel::_GuardedIterator< _RAIter, _Compare >::operator* () [inline]`

Dereference operator.

Returns

Referenced element.

Definition at line 107 of file multiway_merge.h.

5.132.3.3 `template<typename _RAIter, typename _Compare > _GuardedIterator< _RAIter, _Compare >& __gnu_parallel::_GuardedIterator< _RAIter, _Compare >::operator++ () [inline]`

Pre-increment operator.

Returns

This.

Definition at line 98 of file multiway_merge.h.

5.132.4 Friends And Related Function Documentation

5.132.4.1 `template<typename _RAIter, typename _Compare > bool operator< (_GuardedIterator< _RAIter, _Compare > & __bi1, _GuardedIterator< _RAIter, _Compare > & __bi2) [friend]`

Compare two elements referenced by guarded iterators.

Parameters

<code>__bi1</code>	First iterator.
<code>__bi2</code>	Second iterator.

Returns

`true` if less.

Definition at line 120 of file `multiway_merge.h`.

5.132.4.2 `template<typename _RAIter, typename _Compare> bool operator<= (_GuardedIterator<_RAIter,_Compare> & __bi1, _GuardedIterator<_RAIter,_Compare> & __bi2) [friend]`

Compare two elements referenced by guarded iterators.

Parameters

<code>__bi1</code>	First iterator.
<code>__bi2</code>	Second iterator.

Returns

`True` if less equal.

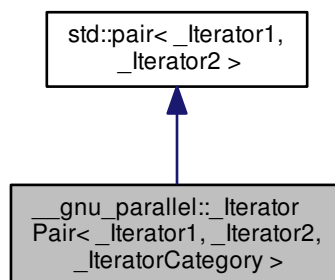
Definition at line 135 of file `multiway_merge.h`.

The documentation for this class was generated from the following file:

- [multiway_merge.h](#)

5.133 `__gnu_parallel::_IteratorPair<_Iterator1,_Iterator2,_IteratorCategory>` Class Template Reference

Inheritance diagram for `__gnu_parallel::_IteratorPair<_Iterator1,_Iterator2,_IteratorCategory>`:



Public Types

- using `_PCCFP` = `_PCC<lis_same< _Iterator1, _U1 >::value||lis_same< _Iterator2, _U2 >::value, _Iterator1, _Iterator2 >`
- using `_PCCP` = `_PCC< true, _Iterator1, _Iterator2 >`
- typedef `std::iterator_traits< _Iterator1 > _TraitsType`
- typedef `_TraitsType::difference_type difference_type`
- typedef `_Iterator1 first_type`
- typedef `_IteratorCategory iterator_category`
- typedef `_IteratorPair * pointer`
- typedef `_IteratorPair & reference`
- typedef `_Iterator2 second_type`
- typedef void `value_type`

Public Member Functions

- `_IteratorPair` (`const _Iterator1 &__first, const _Iterator2 &__second`)
- `operator _Iterator2` () const
- `_IteratorPair operator+` (`difference_type __delta`) const
- `_IteratorPair & operator++` ()
- `const _IteratorPair operator++` (int)
- `difference_type operator-` (`const _IteratorPair &__other`) const
- `_IteratorPair & operator--` ()
- `const _IteratorPair operator--` (int)
- `_IteratorPair & operator=` (`const _IteratorPair &__other`)
- void `swap` (`pair &__p`) noexcept(`__is_nothrow_swappable< _Iterator1 >::value &&__is_nothrow_swappable< _Iterator2 >::value`)

Public Attributes

- `_Iterator1 first`
- `_Iterator2 second`

5.133.1 Detailed Description

```
template<typename _Iterator1, typename _Iterator2, typename _IteratorCategory>
class __gnu_parallel::_IteratorPair< _Iterator1, _Iterator2, _IteratorCategory >
```

A pair of iterators. The usual iterator operations are applied to both child iterators.

Definition at line 45 of file iterator.h.

5.133.2 Member Typedef Documentation

5.133.2.1 using `std::pair< _Iterator1, _Iterator2 >::_PCCFP` = `_PCC<lis_same< _Iterator1, _U1 >::value || lis_same< _Iterator2, _U2 >::value, _Iterator1, _Iterator2 >` [inherited]

There is also a templated copy ctor for the `pair` class itself.

Definition at line 264 of file stl_pair.h.

5.133.2.2 `using std::pair<_Iterator1, _Iterator2>::_PCCP = _PCC<true, _Iterator1, _Iterator2>` [inherited]

Two objects may be passed to a `pair` constructor to be copied.

Definition at line 233 of file `stl_pair.h`.

5.133.2.3 `typedef _Iterator2 std::pair<_Iterator1, _Iterator2>::second_type` [inherited]

`first_type` is the first bound type

Definition at line 193 of file `stl_pair.h`.

5.133.3 Member Data Documentation

5.133.3.1 `_Iterator1 std::pair<_Iterator1, _Iterator2>::first` [inherited]

`second_type` is the second bound type

Definition at line 195 of file `stl_pair.h`.

5.133.3.2 `_Iterator2 std::pair<_Iterator1, _Iterator2>::second` [inherited]

`first` is a copy of the first object

Definition at line 196 of file `stl_pair.h`.

The documentation for this class was generated from the following file:

- [iterator.h](#)

5.134 `__gnu_parallel::_IteratorTriple<_Iterator1, _Iterator2, _Iterator3, _IteratorCategory>` Class Template Reference

Public Types

- `typedef std::iterator_traits<_Iterator1>::difference_type difference_type`
- `typedef _IteratorCategory iterator_category`
- `typedef _IteratorTriple * pointer`
- `typedef _IteratorTriple & reference`
- `typedef void value_type`

Public Member Functions

- **_IteratorTriple** (const _Iterator1 &__first, const _Iterator2 &__second, const _Iterator3 &__third)
- **operator _Iterator3** () const
- **_IteratorTriple operator+** (difference_type __delta) const
- **_IteratorTriple & operator++** ()
- const **_IteratorTriple operator++** (int)
- difference_type **operator-** (const **_IteratorTriple** &__other) const
- **_IteratorTriple & operator--** ()
- const **_IteratorTriple operator--** (int)
- **_IteratorTriple & operator=** (const **_IteratorTriple** &__other)

Public Attributes

- _Iterator1 **_M_first**
- _Iterator2 **_M_second**
- _Iterator3 **_M_third**

5.134.1 Detailed Description

```
template<typename _Iterator1, typename _Iterator2, typename _Iterator3, typename _IteratorCategory>
class __gnu_parallel::_IteratorTriple< _Iterator1, _Iterator2, _Iterator3, _IteratorCategory >
```

A triple of iterators. The usual iterator operations are applied to all three child iterators.

Definition at line 120 of file iterator.h.

The documentation for this class was generated from the following file:

- [iterator.h](#)

5.135 __gnu_parallel::_Job< _DifferenceTp > Struct Template Reference

Public Types

- typedef _DifferenceTp **_DifferenceType**

Public Attributes

- volatile _DifferenceType **_M_first**
- volatile _DifferenceType **_M_last**
- volatile _DifferenceType **_M_load**

5.135.1 Detailed Description

```
template<typename _DifferenceTp>
struct __gnu_parallel::__Job<_DifferenceTp>
```

One `__job` for a certain thread.

Definition at line 54 of file `workstealing.h`.

5.135.2 Member Data Documentation

5.135.2.1 `template<typename _DifferenceTp> volatile _DifferenceType __gnu_parallel::__Job<_DifferenceTp>::_M_first`

First element.

Changed by owning and stealing thread. By stealing thread, always incremented.

Definition at line 62 of file `workstealing.h`.

Referenced by `__gnu_parallel::__for_each_template_random_access_workstealing()`.

5.135.2.2 `template<typename _DifferenceTp> volatile _DifferenceType __gnu_parallel::__Job<_DifferenceTp>::_M_last`

Last element.

Changed by owning thread only.

Definition at line 67 of file `workstealing.h`.

Referenced by `__gnu_parallel::__for_each_template_random_access_workstealing()`.

5.135.2.3 `template<typename _DifferenceTp> volatile _DifferenceType __gnu_parallel::__Job<_DifferenceTp>::_M_load`

Number of elements, i.e. `_M_last-_M_first+1`.

Changed by owning thread only.

Definition at line 72 of file `workstealing.h`.

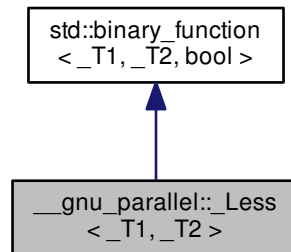
Referenced by `__gnu_parallel::__for_each_template_random_access_workstealing()`.

The documentation for this struct was generated from the following file:

- [workstealing.h](#)

5.136 `__gnu_parallel::_Less<_T1, _T2>` Struct Template Reference

Inheritance diagram for `__gnu_parallel::_Less<_T1, _T2>`:



Public Types

- typedef `_T1` [first_argument_type](#)
- typedef `bool` [result_type](#)
- typedef `_T2` [second_argument_type](#)

Public Member Functions

- `bool operator()` (`const _T1 &__t1, const _T2 &__t2`) `const`
- `bool operator()` (`const _T2 &__t2, const _T1 &__t1`) `const`

5.136.1 Detailed Description

```
template<typename _T1, typename _T2>
struct __gnu_parallel::_Less<_T1, _T2>
```

Similar to `std::less`, but allows two different types.

Definition at line 252 of file `parallel/base.h`.

5.136.2 Member Typedef Documentation

5.136.2.1 typedef `_T1` `std::binary_function<_T1, _T2, bool>::first_argument_type` [\[inherited\]](#)

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.136.2.2 `typedef bool std::binary_function<_T1, _T2, bool>::result_type` [inherited]

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.136.2.3 `typedef _T2 std::binary_function<_T1, _T2, bool>::second_argument_type` [inherited]

`second_argument_type` is the type of the second argument

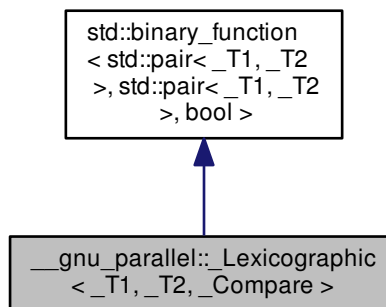
Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [parallel/base.h](#)

5.137 `__gnu_parallel::_Lexicographic<_T1, _T2, _Compare>` Class Template Reference

Inheritance diagram for `__gnu_parallel::_Lexicographic<_T1, _T2, _Compare>`:



Public Types

- `typedef std::pair<_T1, _T2> first_argument_type`
- `typedef bool result_type`
- `typedef std::pair<_T1, _T2> second_argument_type`

Public Member Functions

- `_Lexicographic` (`_Compare` & `__comp`)
- `bool operator()` (const `std::pair<_T1, _T2>` & `__p1`, const `std::pair<_T1, _T2>` & `__p2`) const

5.137.1 Detailed Description

```
template<typename _T1, typename _T2, typename _Compare>
class __gnu_parallel::_Lexicographic< _T1, _T2, _Compare >
```

Compare __a pair of types lexicographically, ascending.

Definition at line 53 of file multiseq_selection.h.

5.137.2 Member Typedef Documentation

5.137.2.1 `typedef std::pair<_T1, _T2> std::binary_function< std::pair<_T1, _T2>, std::pair<_T1, _T2>, bool >::first_argument_type` [inherited]

`first_argument_type` is the type of the first argument

Definition at line 121 of file stl_function.h.

5.137.2.2 `typedef bool std::binary_function< std::pair<_T1, _T2>, std::pair<_T1, _T2>, bool >::result_type` [inherited]

`result_type` is the return type

Definition at line 127 of file stl_function.h.

5.137.2.3 `typedef std::pair<_T1, _T2> std::binary_function< std::pair<_T1, _T2>, std::pair<_T1, _T2>, bool >::second_argument_type` [inherited]

`second_argument_type` is the type of the second argument

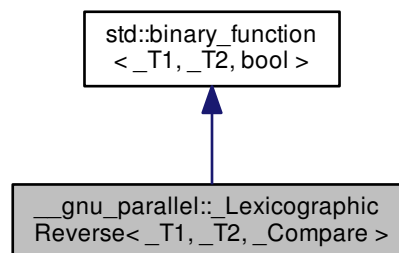
Definition at line 124 of file stl_function.h.

The documentation for this class was generated from the following file:

- [multiseq_selection.h](#)

5.138 __gnu_parallel::_LexicographicReverse<_T1, _T2, _Compare> Class Template Reference

Inheritance diagram for `__gnu_parallel::_LexicographicReverse<_T1, _T2, _Compare>`:



Public Types

- typedef `_T1` [first_argument_type](#)
- typedef `bool` [result_type](#)
- typedef `_T2` [second_argument_type](#)

Public Member Functions

- `_LexicographicReverse` (`_Compare &__comp`)
- `bool operator()` (`const std::pair<_T1, _T2> &__p1, const std::pair<_T1, _T2> &__p2`) `const`

5.138.1 Detailed Description

```
template<typename _T1, typename _T2, typename _Compare>
class __gnu_parallel::_LexicographicReverse<_T1, _T2, _Compare>
```

Compare `__a` pair of types lexicographically, descending.

Definition at line 80 of file `multiseq_selection.h`.

5.138.2 Member Typedef Documentation

5.138.2.1 `typedef _T1 std::binary_function<_T1, _T2, bool>::first_argument_type` `[inherited]`

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.138.2.2 `typedef bool std::binary_function<_T1, _T2, bool>::result_type` `[inherited]`

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.138.2.3 `typedef _T2 std::binary_function<_T1, _T2, bool>::second_argument_type` `[inherited]`

`second_argument_type` is the type of the second argument

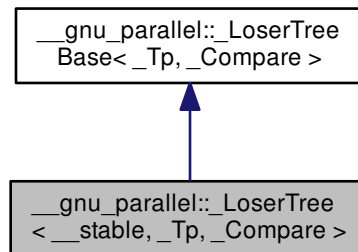
Definition at line 124 of file `stl_function.h`.

The documentation for this class was generated from the following file:

- [multiseq_selection.h](#)

5.139 `__gnu_parallel::_LoserTree< __stable, _Tp, _Compare >` Class Template Reference

Inheritance diagram for `__gnu_parallel::_LoserTree< __stable, _Tp, _Compare >`:



Public Member Functions

- **`_LoserTree`** (unsigned int `__k`, `_Compare` `__comp`)
- void `__delete_min_insert` (`_Tp` `__key`, bool `__sup`)
- int `__get_min_source` ()
- void `__init` ()
- unsigned int `__init_winner` (unsigned int `__root`)
- void `__insert_start` (const `_Tp` & `__key`, int `__source`, bool `__sup`)

Protected Attributes

- `_Compare` `_M_comp`
- bool `_M_first_insert`
- unsigned int `_M_ik`
- unsigned int `_M_k`
- unsigned int `_M_log_k`
- `_Loser` * `_M_losers`
- unsigned int `_M_offset`

5.139.1 Detailed Description

```
template<bool __stable, typename _Tp, typename _Compare>
class __gnu_parallel::_LoserTree< __stable, _Tp, _Compare >
```

Stable `_LoserTree` variant.

Provides the stable implementations of `insert_start`, `__init_winner`, `__init` and `__delete_min_insert`.

Unstable variant is done using partial specialisation below.

Definition at line 169 of file `losertree.h`.

5.139.2 Member Function Documentation

5.139.2.1 `template<bool __stable, typename _Tp, typename _Compare > void __gnu_parallel::_LoserTree< __stable, _Tp, _Compare >::delete_min_insert (_Tp __key, bool __sup) [inline]`

Delete the smallest element and insert a new element from the previously smallest element's sequence.

This implementation is stable.

Definition at line 222 of file `losertree.h`.

References `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::M_comp`, `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser::M_key`, `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::M_losers`, `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser::M_source`, and `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser::M_sup`.

5.139.2.2 `template<typename _Tp, typename _Compare > int __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_get_min_source () [inline],[inherited]`

Returns

the index of the sequence with the smallest element.

Definition at line 155 of file `losertree.h`.

References `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser::M_source`.

5.139.2.3 `template<typename _Tp, typename _Compare > void __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_insert_start (const _Tp & __key, int __source, bool __sup) [inline],[inherited]`

Initializes the sequence "`_M_source`" with the element "`__key`".

Parameters

<code>__key</code>	the element to insert
<code>__source</code>	<code>__index</code> of the <code>__source</code> sequence
<code>__sup</code>	flag that determines whether the value to insert is an explicit <code>__supremum</code> .

Definition at line 134 of file `losertree.h`.

References `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser::M_key`, `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser::M_source`, and `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser::M_sup`.

5.139.3 Member Data Documentation

5.139.3.1 `template<typename _Tp, typename _Compare > _Compare __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_M_comp [protected],[inherited]`

`_Compare` to use.

Definition at line 78 of file losertree.h.

Referenced by `__gnu_parallel::_LoserTree< __stable, _Tp, _Compare >::__delete_min_insert()`, `__gnu_parallel::_LoserTree< false, _Tp, _Compare >::__delete_min_insert()`, and `__gnu_parallel::_LoserTree< false, _Tp, _Compare >::__init_winner()`.

5.139.3.2 `template<typename _Tp, typename _Compare > bool __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::__M_first_insert [protected], [inherited]`

State flag that determines whether the `_LoserTree` is empty.

Only used for building the `_LoserTree`.

Definition at line 85 of file losertree.h.

5.139.3.3 `template<typename _Tp, typename _Compare > unsigned int __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::__M_log_k [protected], [inherited]`

`log_2{__M_k}`

Definition at line 72 of file losertree.h.

Referenced by `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_LoserTreeBase()`.

5.139.3.4 `template<typename _Tp, typename _Compare > _Loser* __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::__M_losers [protected], [inherited]`

`_LoserTree __elements`.

Definition at line 75 of file losertree.h.

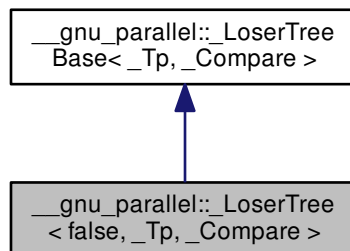
Referenced by `__gnu_parallel::_LoserTree< __stable, _Tp, _Compare >::__delete_min_insert()`, `__gnu_parallel::_LoserTree< false, _Tp, _Compare >::__delete_min_insert()`, `__gnu_parallel::_LoserTree< false, _Tp, _Compare >::__init_winner()`, and `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::~~LoserTreeBase()`.

The documentation for this class was generated from the following file:

- [losertree.h](#)

5.140 `__gnu_parallel::_LoserTree< false, _Tp, _Compare >` Class Template Reference

Inheritance diagram for `__gnu_parallel::_LoserTree< false, _Tp, _Compare >`:



LoserTreeBase< _Tp, _Compare >::_Loser::M_source, and __gnu_parallel::LoserTreeBase< _Tp, _Compare >::_Loser::M_sup.

5.140.2.2 `template<typename _Tp, typename _Compare > int __gnu_parallel::LoserTreeBase< _Tp, _Compare >::_get_min_source() [inline],[inherited]`

Returns

the index of the sequence with the smallest element.

Definition at line 155 of file losertree.h.

References __gnu_parallel::LoserTreeBase< _Tp, _Compare >::_Loser::M_source.

5.140.2.3 `template<typename _Tp, typename _Compare > unsigned int __gnu_parallel::LoserTree< false, _Tp, _Compare >::_init_winner(unsigned int __root) [inline]`

Computes the winner of the competition at position "__root".

Called recursively (starting at 0) to build the initial tree.

Parameters

<code>__root</code>	__index of the "game" to start.
---------------------	---------------------------------

Definition at line 284 of file losertree.h.

References __gnu_parallel::LoserTreeBase< _Tp, _Compare >::_M_comp, __gnu_parallel::LoserTreeBase< _Tp, _Compare >::_Loser::M_key, __gnu_parallel::LoserTreeBase< _Tp, _Compare >::_M_losers, and __gnu_parallel::LoserTreeBase< _Tp, _Compare >::_Loser::M_sup.

5.140.2.4 `template<typename _Tp, typename _Compare > void __gnu_parallel::LoserTreeBase< _Tp, _Compare >::_insert_start(const _Tp & __key, int __source, bool __sup) [inline],[inherited]`

Initializes the sequence "_M_source" with the element "__key".

Parameters

<code>__key</code>	the element to insert
<code>__source</code>	__index of the __source __sequence
<code>__sup</code>	flag that determines whether the value to insert is an explicit __supremum.

Definition at line 134 of file losertree.h.

References __gnu_parallel::LoserTreeBase< _Tp, _Compare >::_Loser::M_key, __gnu_parallel::LoserTreeBase< _Tp, _Compare >::_Loser::M_source, and __gnu_parallel::LoserTreeBase< _Tp, _Compare >::_Loser::M_sup.

5.140.3 Member Data Documentation

5.140.3.1 `template<typename _Tp, typename _Compare > _Compare __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_M_comp` [protected], [inherited]

`_Compare` to use.

Definition at line 78 of file `losertree.h`.

Referenced by `__gnu_parallel::_LoserTree< __stable, _Tp, _Compare >::_delete_min_insert()`, `__delete_min_↵insert()`, and `__init_winner()`.

5.140.3.2 `template<typename _Tp, typename _Compare > bool __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_M_first_insert` [protected], [inherited]

State flag that determines whether the `_LoserTree` is empty.

Only used for building the `_LoserTree`.

Definition at line 85 of file `losertree.h`.

5.140.3.3 `template<typename _Tp, typename _Compare > unsigned int __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_M_log_k` [protected], [inherited]

`log_2(_M_k)`

Definition at line 72 of file `losertree.h`.

Referenced by `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_LoserTreeBase()`.

5.140.3.4 `template<typename _Tp, typename _Compare > _Loser* __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_M_losers` [protected], [inherited]

`_LoserTree` `__elements`.

Definition at line 75 of file `losertree.h`.

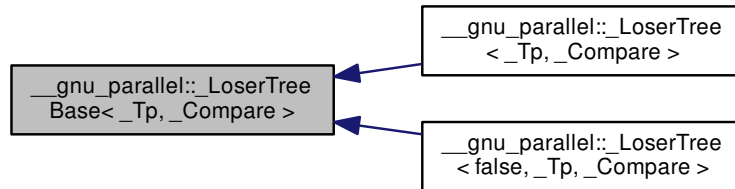
Referenced by `__gnu_parallel::_LoserTree< __stable, _Tp, _Compare >::_delete_min_insert()`, `__delete_min_↵insert()`, `__init_winner()`, and `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::~~_LoserTreeBase()`.

The documentation for this class was generated from the following file:

- [losertree.h](#)

5.141 `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >` Class Template Reference

Inheritance diagram for `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >`:



Classes

- struct [_Loser](#)

Public Member Functions

- [_LoserTreeBase](#) (unsigned int __k, _Compare __comp)
- [~_LoserTreeBase](#) ()
- int [__get_min_source](#) ()
- void [__insert_start](#) (const _Tp &__key, int __source, bool __sup)

Protected Attributes

- [_Compare _M_comp](#)
- bool [_M_first_insert](#)
- unsigned int [_M_ik](#)
- unsigned int [_M_k](#)
- unsigned int [_M_log_k](#)
- [_Loser * _M_losers](#)
- unsigned int [_M_offset](#)

5.141.1 Detailed Description

```
template<typename _Tp, typename _Compare>
class __gnu_parallel::_LoserTreeBase< _Tp, _Compare >
```

Guarded loser/tournament tree.

The smallest element is at the top.

Guarding is done explicitly through one flag `_M_sup` per element, `inf` is not needed due to a better initialization routine. This is a well-performing variant.

Parameters

<code>_Tp</code>	the element type
<code>_Compare</code>	the comparator to use, defaults to <code>std::less<_Tp></code>

Definition at line 55 of file `losertree.h`.

5.141.2 Constructor & Destructor Documentation

5.141.2.1 `template<typename _Tp, typename _Compare> __gnu_parallel::_LoserTreeBase<_Tp, _Compare>::_LoserTreeBase(unsigned int __k, _Compare __comp) [inline]`

The constructor.

Parameters

<code>__k</code>	The number of sequences to merge.
<code>__comp</code>	The comparator to use.

Definition at line 94 of file `losertree.h`.

References `__gnu_parallel::_rd_log2()`, `__gnu_parallel::_LoserTreeBase<_Tp, _Compare>::_M_log_k`, and `__gnu_parallel::_LoserTreeBase<_Tp, _Compare>::_Loser::_M_sup`.

5.141.2.2 `template<typename _Tp, typename _Compare> __gnu_parallel::_LoserTreeBase<_Tp, _Compare>::~~_LoserTreeBase() [inline]`

The destructor.

Definition at line 118 of file `losertree.h`.

References `__gnu_parallel::_LoserTreeBase<_Tp, _Compare>::_M_losers`.

5.141.3 Member Function Documentation

5.141.3.1 `template<typename _Tp, typename _Compare> int __gnu_parallel::_LoserTreeBase<_Tp, _Compare>::_get_min_source() [inline]`

Returns

the index of the sequence with the smallest element.

Definition at line 155 of file `losertree.h`.

References `__gnu_parallel::_LoserTreeBase<_Tp, _Compare>::_Loser::_M_source`.

5.141.3.2 `template<typename _Tp, typename _Compare> void __gnu_parallel::_LoserTreeBase<_Tp, _Compare>::_insert_start(const _Tp & __key, int __source, bool __sup) [inline]`

Initializes the sequence "`_M_source`" with the element "`__key`".

Parameters

<code>__key</code>	the element to insert
<code>__source</code>	<code>__index</code> of the <code>__source</code> <code>__sequence</code>
<code>__sup</code>	flag that determines whether the value to insert is an explicit <code>__supremum</code> .

Definition at line 134 of file `losertree.h`.

References `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser::_M_key`, `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser::_M_source`, and `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser::_M_sup`.

5.141.4 Member Data Documentation

5.141.4.1 `template<typename _Tp, typename _Compare > _Compare __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_M_comp` `[protected]`

`_Compare` to use.

Definition at line 78 of file `losertree.h`.

Referenced by `__gnu_parallel::_LoserTree< __stable, _Tp, _Compare >::_delete_min_insert()`, `__gnu_parallel::_LoserTree< false, _Tp, _Compare >::_delete_min_insert()`, and `__gnu_parallel::_LoserTree< false, _Tp, _Compare >::_init_winner()`.

5.141.4.2 `template<typename _Tp, typename _Compare > bool __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_M_first_insert` `[protected]`

State flag that determines whether the `_LoserTree` is empty.

Only used for building the `_LoserTree`.

Definition at line 85 of file `losertree.h`.

5.141.4.3 `template<typename _Tp, typename _Compare > unsigned int __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_M_log_k` `[protected]`

`log_2(_M_k)`

Definition at line 72 of file `losertree.h`.

Referenced by `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_LoserTreeBase()`.

5.141.4.4 `template<typename _Tp, typename _Compare> _Loser* __gnu_parallel::_LoserTreeBase<_Tp, _Compare>::_M_losers` [protected]

`_LoserTree` __elements.

Definition at line 75 of file `losertree.h`.

Referenced by `__gnu_parallel::_LoserTree<__stable, _Tp, _Compare>::__delete_min_insert()`, `__gnu_parallel::_LoserTree<false, _Tp, _Compare>::__delete_min_insert()`, `__gnu_parallel::_LoserTree<false, _Tp, _Compare>::__init_winner()`, and `__gnu_parallel::_LoserTreeBase<_Tp, _Compare>::~~_LoserTreeBase()`.

The documentation for this class was generated from the following file:

- [losertree.h](#)

5.142 `__gnu_parallel::_LoserTreeBase<_Tp, _Compare>::_Loser` Struct Reference

Public Attributes

- `_Tp` [_M_key](#)
- `int` [_M_source](#)
- `bool` [_M_sup](#)

5.142.1 Detailed Description

```
template<typename _Tp, typename _Compare>
struct __gnu_parallel::_LoserTreeBase<_Tp, _Compare>::_Loser
```

Internal representation of a `_LoserTree` element.

Definition at line 59 of file `losertree.h`.

5.142.2 Member Data Documentation

5.142.2.1 `template<typename _Tp, typename _Compare> _Tp __gnu_parallel::_LoserTreeBase<_Tp, _Compare>::_Loser::_M_key`

`_M_key` of the element in the `_LoserTree`.

Definition at line 66 of file `losertree.h`.

Referenced by `__gnu_parallel::_LoserTree<__stable, _Tp, _Compare>::__delete_min_insert()`, `__gnu_parallel::_LoserTree<false, _Tp, _Compare>::__delete_min_insert()`, `__gnu_parallel::_LoserTree<false, _Tp, _Compare>::__init_winner()`, and `__gnu_parallel::_LoserTreeBase<_Tp, _Compare>::__insert_start()`.

5.142.2.2 `template<typename _Tp, typename _Compare > int __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser::_M_source`

__index of the __source __sequence.

Definition at line 64 of file losertree.h.

Referenced by `__gnu_parallel::_LoserTree< __stable, _Tp, _Compare >::_delete_min_insert()`, `__gnu_parallel::_LoserTree< false, _Tp, _Compare >::_delete_min_insert()`, `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_get_min_source()`, and `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_insert_start()`.

5.142.2.3 `template<typename _Tp, typename _Compare > bool __gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_Loser::_M_sup`

flag, true iff this is a "maximum" __sentinel.

Definition at line 62 of file losertree.h.

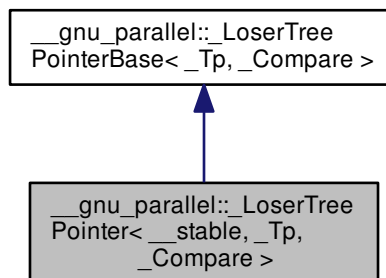
Referenced by `__gnu_parallel::_LoserTree< __stable, _Tp, _Compare >::_delete_min_insert()`, `__gnu_parallel::_LoserTree< false, _Tp, _Compare >::_delete_min_insert()`, `__gnu_parallel::_LoserTree< false, _Tp, _Compare >::_init_winner()`, `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_insert_start()`, and `__gnu_parallel::_LoserTreeBase< _Tp, _Compare >::_LoserTreeBase()`.

The documentation for this struct was generated from the following file:

- [losertree.h](#)

5.143 `__gnu_parallel::_LoserTreePointer< __stable, _Tp, _Compare >` Class Template Reference

Inheritance diagram for `__gnu_parallel::_LoserTreePointer< __stable, _Tp, _Compare >`:



Public Member Functions

- `_LoserTreePointer` (unsigned int __k, _Compare __comp=[std::less](#)< _Tp >())
- void `__delete_min_insert` (const _Tp &__key, bool __sup)
- int `__get_min_source` ()
- void `__init` ()
- unsigned int `__init_winner` (unsigned int __root)
- void `__insert_start` (const _Tp &__key, int __source, bool __sup)

Protected Attributes

- _Compare `_M_comp`
- unsigned int `_M_ik`
- unsigned int `_M_k`
- [_Loser](#) * `_M_losers`
- unsigned int `_M_offset`

5.143.1 Detailed Description

```
template<bool __stable, typename _Tp, typename _Compare>
class __gnu_parallel::_LoserTreePointer< __stable, _Tp, _Compare >
```

Stable `_LoserTree` implementation.

The unstable variant is implemented using partial instantiation below.

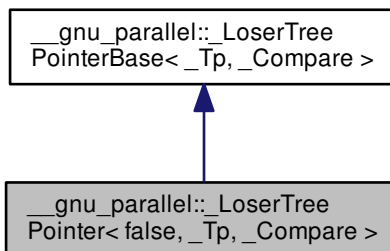
Definition at line 409 of file `losertree.h`.

The documentation for this class was generated from the following file:

- [losertree.h](#)

5.144 `__gnu_parallel::_LoserTreePointer< false, _Tp, _Compare >` Class Template Reference

Inheritance diagram for `__gnu_parallel::_LoserTreePointer< false, _Tp, _Compare >`:



Public Member Functions

- **_LoserTreePointer** (unsigned int __k, _Compare __comp=[std::less](#)< _Tp >())
- void **__delete_min_insert** (const _Tp &__key, bool __sup)
- int **__get_min_source** ()
- void **__init** ()
- unsigned int **__init_winner** (unsigned int __root)
- void **__insert_start** (const _Tp &__key, int __source, bool __sup)

Protected Attributes

- _Compare **_M_comp**
- unsigned int **_M_ik**
- unsigned int **_M_k**
- [_Loser](#) * **_M_losers**
- unsigned int **_M_offset**

5.144.1 Detailed Description

```
template<typename _Tp, typename _Compare>
class __gnu_parallel::_LoserTreePointer< false, _Tp, _Compare >
```

Unstable _LoserTree implementation.

The stable variant is above.

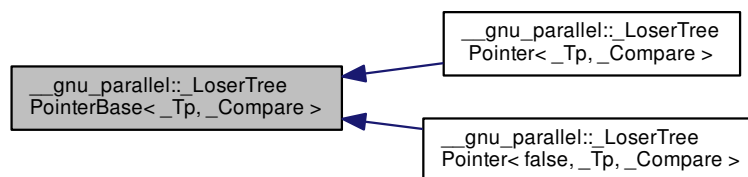
Definition at line 491 of file losertree.h.

The documentation for this class was generated from the following file:

- [losertree.h](#)

5.145 __gnu_parallel::_LoserTreePointerBase< _Tp, _Compare > Class Template Reference

Inheritance diagram for __gnu_parallel::_LoserTreePointerBase< _Tp, _Compare >:



Classes

- struct [_Loser](#)

Public Member Functions

- `_LoserTreePointerBase` (unsigned int __k, _Compare __comp=[std::less](#)< _Tp >())
- int `__get_min_source` ()
- void `__insert_start` (const _Tp &__key, int __source, bool __sup)

Protected Attributes

- _Compare `_M_comp`
- unsigned int `_M_ik`
- unsigned int `_M_k`
- [_Loser](#) * `_M_losers`
- unsigned int `_M_offset`

5.145.1 Detailed Description

```
template<typename _Tp, typename _Compare>
class __gnu_parallel::_LoserTreePointerBase< _Tp, _Compare >
```

Base class of `_Loser` Tree implementation using pointers.

Definition at line 357 of file `losertree.h`.

The documentation for this class was generated from the following file:

- [losertree.h](#)

5.146 `__gnu_parallel::_LoserTreePointerBase< _Tp, _Compare >::_Loser` Struct Reference

Public Attributes

- const _Tp * `_M_keyp`
- int `_M_source`
- bool `_M_sup`

5.146.1 Detailed Description

```
template<typename _Tp, typename _Compare>
struct __gnu_parallel::_LoserTreePointerBase< _Tp, _Compare >::_Loser
```

Internal representation of `_LoserTree` __elements.

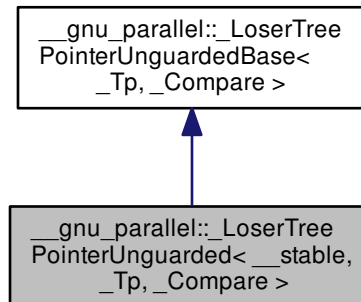
Definition at line 361 of file `losertree.h`.

The documentation for this struct was generated from the following file:

- [losertree.h](#)

5.147 `__gnu_parallel::_LoserTreePointerUnguarded< __stable, _Tp, _Compare >` Class Template Reference

Inheritance diagram for `__gnu_parallel::_LoserTreePointerUnguarded< __stable, _Tp, _Compare >`:



Public Member Functions

- **`_LoserTreePointerUnguarded`** (unsigned int __k, const _Tp &__sentinel, _Compare __comp=`std::less< _Tp >()`)
- void **`__delete_min_insert`** (const _Tp &__key, bool __sup)
- int **`__get_min_source`** ()
- void **`__init`** ()
- unsigned int **`__init_winner`** (unsigned int __root)
- void **`__insert_start`** (const _Tp &__key, int __source, bool)

Protected Attributes

- `_Compare` **`_M_comp`**
- unsigned int **`_M_ik`**
- unsigned int **`_M_k`**
- `_Loser` * **`_M_losers`**
- unsigned int **`_M_offset`**

5.147.1 Detailed Description

```
template<bool __stable, typename _Tp, typename _Compare>  
class __gnu_parallel::_LoserTreePointerUnguarded< __stable, _Tp, _Compare >
```

Stable unguarded `_LoserTree` variant storing pointers.

Unstable variant is implemented below using partial specialization.

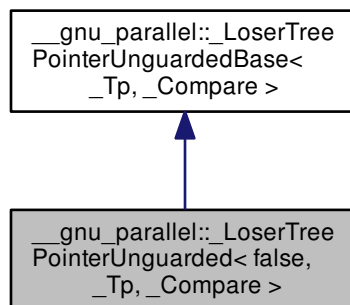
Definition at line 891 of file `losertree.h`.

The documentation for this class was generated from the following file:

- [losertree.h](#)

5.148 `__gnu_parallel::_LoserTreePointerUnguarded< false, _Tp, _Compare >` Class Template Reference

Inheritance diagram for `__gnu_parallel::_LoserTreePointerUnguarded< false, _Tp, _Compare >`:



Public Member Functions

- **_LoserTreePointerUnguarded** (unsigned int __k, const _Tp &__sentinel, _Compare __comp=[std::less](#)< _Tp >())
- void **__delete_min_insert** (const _Tp &__key, bool __sup)
- int **__get_min_source** ()
- void **__init** ()
- unsigned int **__init_winner** (unsigned int __root)
- void **__insert_start** (const _Tp &__key, int __source, bool)

Protected Attributes

- _Compare **_M_comp**
- unsigned int **_M_ik**
- unsigned int **_M_k**
- _Loser * **_M_losers**
- unsigned int **_M_offset**

5.148.1 Detailed Description

```
template<typename _Tp, typename _Compare>
class __gnu_parallel::_LoserTreePointerUnguarded< false, _Tp, _Compare >
```

Unstable unguarded _LoserTree variant storing pointers.

Stable variant is above.

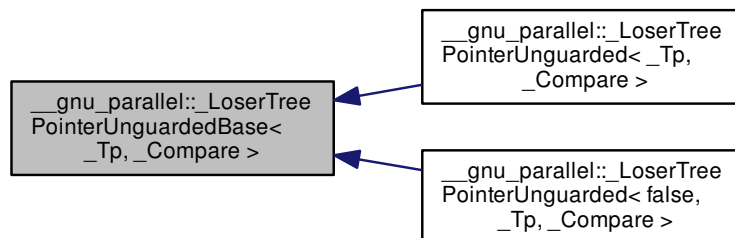
Definition at line 977 of file losertree.h.

The documentation for this class was generated from the following file:

- [losertree.h](#)

5.149 __gnu_parallel::_LoserTreePointerUnguardedBase< _Tp, _Compare > Class Template Reference

Inheritance diagram for __gnu_parallel::_LoserTreePointerUnguardedBase< _Tp, _Compare >:



Public Member Functions

- `_LoserTreePointerUnguardedBase` (unsigned int __k, const _Tp &__sentinel, _Compare __comp=[std::less<_Tp>](#)())
- int `__get_min_source` ()
- void `__insert_start` (const _Tp &__key, int __source, bool)

Protected Attributes

- _Compare `_M_comp`
- unsigned int `_M_ik`
- unsigned int `_M_k`
- _Loser * `_M_losers`
- unsigned int `_M_offset`

5.149.1 Detailed Description

```
template<typename _Tp, typename _Compare>
class __gnu_parallel::_LoserTreePointerUnguardedBase<_Tp, _Compare >
```

Unguarded loser tree, keeping only pointers to the elements in the tree structure.

No guarding is done, therefore not a single input sequence must run empty. This is a very fast variant.

Definition at line 828 of file `losertree.h`.

The documentation for this class was generated from the following file:

- [losertree.h](#)

5.150 `__gnu_parallel::_LoserTreeTraits<_Tp>` Struct Template Reference

Static Public Attributes

- static const bool `_M_use_pointer`

5.150.1 Detailed Description

```
template<typename _Tp>
struct __gnu_parallel::_LoserTreeTraits<_Tp >
```

Traits for determining whether the loser tree should use pointers or copies.

The field "`_M_use_pointer`" is used to determine whether to use pointers in the loser trees or whether to copy the values into the loser tree.

The default behavior is to use pointers if the data type is 4 times as big as the pointer to it.

Specialize for your data type to customize the behavior.

Example:

```
template<> struct _LoserTreeTraits<int> { static const bool _M_use_pointer = false; };
template<> struct _LoserTreeTraits<heavyweight_type> { static const bool _M_use_pointer = true; };
```

Parameters

<code>_Tp</code>	type to give the loser tree traits for.
------------------	---

Definition at line 731 of file multiway_merge.h.

5.150.2 Member Data Documentation

5.150.2.1 `template<typename _Tp> const bool __gnu_parallel::_LoserTreeTraits<_Tp>::_M_use_pointer` [static]

True iff to use pointers instead of values in loser trees.

The default behavior is to use pointers if the data type is four times as big as the pointer to it.

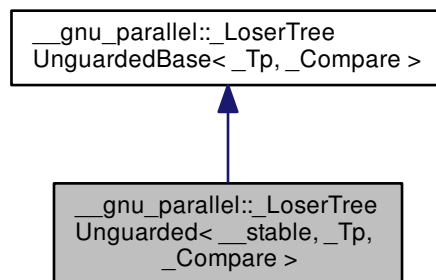
Definition at line 739 of file multiway_merge.h.

The documentation for this struct was generated from the following file:

- [multiway_merge.h](#)

5.151 `__gnu_parallel::_LoserTreeUnguarded<__stable, _Tp, _Compare>` Class Template Reference

Inheritance diagram for `__gnu_parallel::_LoserTreeUnguarded<__stable, _Tp, _Compare>`:



Public Member Functions

- **`_LoserTreeUnguarded`** (unsigned int __k, const _Tp &__sentinel, _Compare __comp=[std::less](#)<_Tp>())
- void **`__delete_min_insert`** (_Tp __key, bool)
- int **`__get_min_source`** ()
- void **`__init`** ()
- unsigned int **`__init_winner`** (unsigned int __root)
- void **`__insert_start`** (const _Tp &__key, int __source, bool)

Protected Attributes

- `_Compare` **_M_comp**
- unsigned int **_M_ik**
- unsigned int **_M_k**
- `_Loser` * **_M_losers**
- unsigned int **_M_offset**

5.151.1 Detailed Description

```
template<bool __stable, typename _Tp, typename _Compare>
class __gnu_parallel::_LoserTreeUnguarded< __stable, _Tp, _Compare >
```

Stable implementation of unguarded `_LoserTree`.

Unstable variant is selected below with partial specialization.

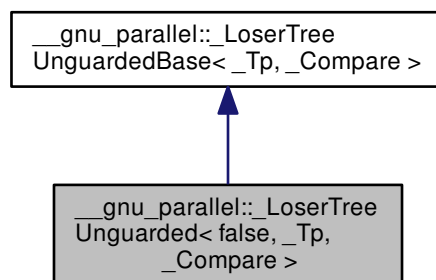
Definition at line 646 of file `losertree.h`.

The documentation for this class was generated from the following file:

- [losertree.h](#)

5.152 `__gnu_parallel::_LoserTreeUnguarded< false, _Tp, _Compare >` Class Template Reference

Inheritance diagram for `__gnu_parallel::_LoserTreeUnguarded< false, _Tp, _Compare >`:



Public Member Functions

- **_LoserTreeUnguarded** (unsigned int __k, const _Tp &__sentinel, _Compare __comp=[std::less](#)< _Tp >())
- void **__delete_min_insert** (_Tp __key, bool)
- int **__get_min_source** ()
- void **__init** ()
- unsigned int **__init_winner** (unsigned int __root)
- void **__insert_start** (const _Tp &__key, int __source, bool)

Protected Attributes

- _Compare **_M_comp**
- unsigned int **_M_ik**
- unsigned int **_M_k**
- _Loser * **_M_losers**
- unsigned int **_M_offset**

5.152.1 Detailed Description

```
template<typename _Tp, typename _Compare>
class __gnu_parallel::_LoserTreeUnguarded< false, _Tp, _Compare >
```

Non-Stable implementation of unguarded _LoserTree.

Stable implementation is above.

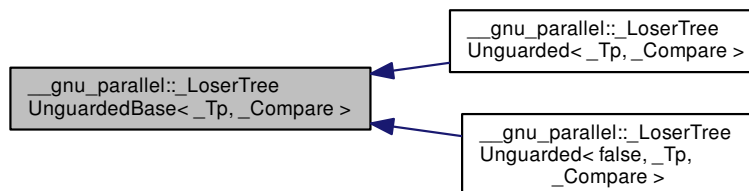
Definition at line 734 of file losertree.h.

The documentation for this class was generated from the following file:

- [losertree.h](#)

5.153 __gnu_parallel::_LoserTreeUnguardedBase< _Tp, _Compare > Class Template Reference

Inheritance diagram for __gnu_parallel::_LoserTreeUnguardedBase< _Tp, _Compare >:



Public Member Functions

- `_LoserTreeUnguardedBase` (unsigned int __k, const _Tp &__sentinel, _Compare __comp=[std::less<_Tp>\(\)](#))
- `int __get_min_source` ()
- `void __insert_start` (const _Tp &__key, int __source, bool)

Protected Attributes

- `_Compare _M_comp`
- `unsigned int _M_ik`
- `unsigned int _M_k`
- `_Loser * _M_losers`
- `unsigned int _M_offset`

5.153.1 Detailed Description

```
template<typename _Tp, typename _Compare>
class __gnu_parallel::_LoserTreeUnguardedBase<_Tp, _Compare>
```

Base class for unguarded `_LoserTree` implementation.

The whole element is copied into the tree structure.

No guarding is done, therefore not a single input sequence must run empty. Unused `__sequence` heads are marked with a sentinel which is `>` all elements that are to be merged.

This is a very fast variant.

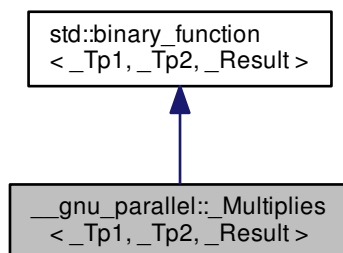
Definition at line 574 of file `losertree.h`.

The documentation for this class was generated from the following file:

- [losertree.h](#)

5.154 `__gnu_parallel::_Multiplies<_Tp1, _Tp2, _Result>` Struct Template Reference

Inheritance diagram for `__gnu_parallel::_Multiplies<_Tp1, _Tp2, _Result>`:



Public Types

- typedef `_Tp1` [first_argument_type](#)
- typedef `_Result` [result_type](#)
- typedef `_Tp2` [second_argument_type](#)

Public Member Functions

- `_Result` **operator()** (const `_Tp1` &__x, const `_Tp2` &__y) const

5.154.1 Detailed Description

```
template<typename _Tp1, typename _Tp2, typename _Result = __typeof__(*static_cast<_Tp1*>(0) * *static_cast<_Tp2*>(0))>
struct __gnu_parallel::__Multiplies<_Tp1, _Tp2, _Result >
```

Similar to `std::multiplies`, but allows two different types.

Definition at line 288 of file `parallel/base.h`.

5.154.2 Member Typedef Documentation

5.154.2.1 `typedef _Tp1 std::binary_function<_Tp1, _Tp2, _Result>::first_argument_type` `[inherited]`

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.154.2.2 `typedef _Result std::binary_function<_Tp1, _Tp2, _Result>::result_type` `[inherited]`

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.154.2.3 `typedef _Tp2 std::binary_function<_Tp1, _Tp2, _Result>::second_argument_type` `[inherited]`

`second_argument_type` is the type of the second argument

Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [parallel/base.h](#)

5.155 `__gnu_parallel::_Nothing` Struct Reference

Public Member Functions

- `template<typename _It>`
`void operator\(\) (_It __i)`

5.155.1 Detailed Description

Functor doing nothing.

For some `__reduction` tasks (this is not a function object, but is passed as `__selector` `__dummy` parameter.

Definition at line 288 of file `for_each_selectors.h`.

5.155.2 Member Function Documentation

5.155.2.1 `template<typename _It> void __gnu_parallel::_Nothing::operator() (_It __i)` `[inline]`

Functor execution.

Parameters

<code>↔</code>	iterator referencing object.
<code>↔</code>	
<code>↔</code>	
<code>↔</code>	
<code>↔</code>	
<code>i</code>	

Definition at line 294 of file `for_each_selectors.h`.

The documentation for this struct was generated from the following file:

- [for_each_selectors.h](#)

5.156 `__gnu_parallel::_Piece<_DifferenceTp>` Struct Template Reference

Public Types

- `typedef _DifferenceTp _DifferenceType`

Public Attributes

- `_DifferenceType _M_begin`
- `_DifferenceType _M_end`

5.156.1 Detailed Description

```
template<typename _DifferenceTp>
struct __gnu_parallel::_Piece< _DifferenceTp >
```

Subsequence description.

Definition at line 46 of file multiway_mergesort.h.

5.156.2 Member Data Documentation

5.156.2.1 `template<typename _DifferenceTp > _DifferenceType __gnu_parallel::_Piece< _DifferenceTp >::_M_begin`

Begin of subsequence.

Definition at line 51 of file multiway_mergesort.h.

5.156.2.2 `template<typename _DifferenceTp > _DifferenceType __gnu_parallel::_Piece< _DifferenceTp >::_M_end`

End of subsequence.

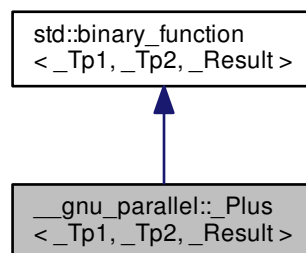
Definition at line 54 of file multiway_mergesort.h.

The documentation for this struct was generated from the following file:

- [multiway_mergesort.h](#)

5.157 __gnu_parallel::_Plus< _Tp1, _Tp2, _Result > Struct Template Reference

Inheritance diagram for `__gnu_parallel::_Plus< _Tp1, _Tp2, _Result >`:



Public Types

- typedef `_Tp1` [first_argument_type](#)
- typedef `_Result` [result_type](#)
- typedef `_Tp2` [second_argument_type](#)

Public Member Functions

- `_Result` **operator()** (const `_Tp1` &__x, const `_Tp2` &__y) const

5.157.1 Detailed Description

```
template<typename _Tp1, typename _Tp2, typename _Result = __typeof__(*static_cast<_Tp1*>(0) + *static_cast<_Tp2*>(0))>
struct __gnu_parallel::__Plus<_Tp1, _Tp2, _Result >
```

Similar to `std::plus`, but allows two different types.

Definition at line 272 of file `parallel/base.h`.

5.157.2 Member Typedef Documentation

5.157.2.1 `typedef _Tp1 std::binary_function<_Tp1, _Tp2, _Result >::first_argument_type` `[inherited]`

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.157.2.2 `typedef _Result std::binary_function<_Tp1, _Tp2, _Result >::result_type` `[inherited]`

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.157.2.3 `typedef _Tp2 std::binary_function<_Tp1, _Tp2, _Result >::second_argument_type` `[inherited]`

`second_argument_type` is the type of the second argument

Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [parallel/base.h](#)

5.158 `__gnu_parallel::PMWMSortingData<_RAIter>` Struct Template Reference

Public Types

- typedef `_TraitsType::difference_type` **`_DifferenceType`**
- typedef `std::iterator_traits<_RAIter>` **`_TraitsType`**
- typedef `_TraitsType::value_type` **`_ValueType`**

Public Attributes

- `_ThreadIndex` `_M_num_threads`
- `_DifferenceType` * `_M_offsets`
- `std::vector<_Piece<_DifferenceType>>` * `_M_pieces`
- `_ValueType` * `_M_samples`
- `_RAIter` `_M_source`
- `_DifferenceType` * `_M_starts`
- `_ValueType` ** `_M_temporary`

5.158.1 Detailed Description

```
template<typename _RAIter>
struct __gnu_parallel::PMWMSortingData<_RAIter>
```

Data accessed by all threads.

PMWMS = parallel multiway mergesort

Definition at line 61 of file `multiway_mergesort.h`.

5.158.2 Member Data Documentation

5.158.2.1 `template<typename _RAIter> _ThreadIndex __gnu_parallel::PMWMSortingData<_RAIter>::_M_num_threads`

Number of threads involved.

Definition at line 68 of file `multiway_mergesort.h`.

Referenced by `__gnu_parallel::parallel_sort_mwms()`, and `__gnu_parallel::parallel_sort_mwms_pu()`.

5.158.2.2 `template<typename _RAIter> _DifferenceType* __gnu_parallel::PMWMSortingData<_RAIter>::_M_offsets`

Offsets to add to the found positions.

Definition at line 83 of file `multiway_mergesort.h`.

Referenced by `__gnu_parallel::parallel_sort_mwms()`.

5.158.2.3 `template<typename _RAIter> std::vector<_Piece<_DifferenceType>> * __gnu_parallel::PMWMSortingData<_RAIter>::M_pieces`

Pieces of data to merge [thread][__sequence].

Definition at line 86 of file `multiway_mergesort.h`.

Referenced by `__gnu_parallel::parallel_sort_mwms()`, and `__gnu_parallel::parallel_sort_mwms_pu()`.

5.158.2.4 `template<typename _RAIter> _ValueType* __gnu_parallel::PMWMSortingData<_RAIter>::M_samples`

Samples.

Definition at line 80 of file `multiway_mergesort.h`.

Referenced by `__gnu_parallel::__determine_samples()`, and `__gnu_parallel::parallel_sort_mwms()`.

5.158.2.5 `template<typename _RAIter> _RAIter __gnu_parallel::PMWMSortingData<_RAIter>::M_source`

Input __begin.

Definition at line 71 of file `multiway_mergesort.h`.

Referenced by `__gnu_parallel::__determine_samples()`, `__gnu_parallel::parallel_sort_mwms()`, and `__gnu_parallel::parallel_sort_mwms_pu()`.

5.158.2.6 `template<typename _RAIter> _DifferenceType* __gnu_parallel::PMWMSortingData<_RAIter>::M_starts`

Start indices, per thread.

Definition at line 74 of file `multiway_mergesort.h`.

Referenced by `__gnu_parallel::__determine_samples()`, `__gnu_parallel::parallel_sort_mwms()`, and `__gnu_parallel::parallel_sort_mwms_pu()`.

5.158.2.7 `template<typename _RAIter> _ValueType** __gnu_parallel::PMWMSortingData<_RAIter>::M_temporary`

Storage in which to sort.

Definition at line 77 of file `multiway_mergesort.h`.

Referenced by `__gnu_parallel::parallel_sort_mwms()`, and `__gnu_parallel::parallel_sort_mwms_pu()`.

The documentation for this struct was generated from the following file:

- [multiway_mergesort.h](#)

5.159 `__gnu_parallel::_PseudoSequence<_Tp, _DifferenceTp>` Class Template Reference

Public Types

- typedef `_DifferenceTp` **`_DifferenceType`**
- typedef `_PseudoSequenceIterator<_Tp, uint64_t>` **`iterator`**

Public Member Functions

- `_PseudoSequence` (const `_Tp` & `__val`, `_DifferenceType` `__count`)
- `iterator begin` () const
- `iterator end` () const

5.159.1 Detailed Description

```
template<typename _Tp, typename _DifferenceTp>
class __gnu_parallel::_PseudoSequence<_Tp, _DifferenceTp>
```

Sequence that conceptually consists of multiple copies of the same element. The copies are not stored explicitly, of course.

Parameters

<code>_Tp</code>	Sequence <code>_M_value</code> type.
<code>_DifferenceTp</code>	Sequence difference type.

Definition at line 359 of file `parallel/base.h`.

5.159.2 Constructor & Destructor Documentation

5.159.2.1 `template<typename _Tp, typename _DifferenceTp> __gnu_parallel::_PseudoSequence<_Tp, _DifferenceTp>::__PseudoSequence (const _Tp & __val, _DifferenceType __count)` `[inline]`

Constructor.

Parameters

<code>__val</code>	Element of the sequence.
<code>__count</code>	Number of (virtual) copies.

Definition at line 371 of file `parallel/base.h`.

5.159.3 Member Function Documentation

5.159.3.1 `template<typename _Tp, typename _DifferenceTp> iterator __gnu_parallel::_PseudoSequence<_Tp, _DifferenceTp>::begin () const [inline]`

Begin iterator.

Definition at line 376 of file `parallel/base.h`.

5.159.3.2 `template<typename _Tp, typename _DifferenceTp> iterator __gnu_parallel::_PseudoSequence<_Tp, _DifferenceTp>::end () const [inline]`

End iterator.

Definition at line 381 of file `parallel/base.h`.

The documentation for this class was generated from the following file:

- [parallel/base.h](#)

5.160 `__gnu_parallel::_PseudoSequenceliterator<_Tp, _DifferenceTp>` Class Template Reference

Public Types

- `typedef _DifferenceTp _DifferenceType`

Public Member Functions

- `_PseudoSequenceliterator (const _Tp &__val, _DifferenceType __pos)`
- `bool operator!= (const _PseudoSequenceliterator &__i2)`
- `const _Tp & operator* () const`
- `_PseudoSequenceliterator & operator++ ()`
- `_PseudoSequenceliterator operator++ (int)`
- `_DifferenceType operator- (const _PseudoSequenceliterator &__i2)`
- `bool operator== (const _PseudoSequenceliterator &__i2)`
- `const _Tp & operator[] (_DifferenceType) const`

5.160.1 Detailed Description

```
template<typename _Tp, typename _DifferenceTp>
class __gnu_parallel::_PseudoSequenceliterator<_Tp, _DifferenceTp>
```

`_Iterator` associated with `__gnu_parallel::_PseudoSequence`. It features the usual random-access iterator functionality.

Parameters

<code>_Tp</code>	Sequence <code>_M_value</code> type.
<code>_DifferenceTp</code>	Sequence difference type.

Definition at line 306 of file parallel/base.h.

The documentation for this class was generated from the following file:

- [parallel/base.h](#)

5.161 `__gnu_parallel::__QSBThreadLocal<_RAIter >` Struct Template Reference

Public Types

- typedef `_TraitsType::difference_type` **_DifferenceType**
- typedef `std::pair<_RAIter, _RAIter >` **_Piece**
- typedef `std::iterator_traits<_RAIter >` **_TraitsType**

Public Member Functions

- **`_QSBThreadLocal`** (int `__queue_size`)

Public Attributes

- volatile `_DifferenceType *` **`_M_elements_leftover`**
- **`_Piece`** **`_M_global`**
- **`_Piece`** **`_M_initial`**
- **`_RestrictedBoundedConcurrentQueue<_Piece >`** **`_M_leftover_parts`**
- **`_ThreadIndex`** **`_M_num_threads`**

5.161.1 Detailed Description

```
template<typename _RAIter>
struct __gnu_parallel::__QSBThreadLocal<_RAIter >
```

Information local to one thread in the parallel quicksort run.

Definition at line 65 of file `balanced_quicksort.h`.

5.161.2 Member Typedef Documentation

5.161.2.1 `template<typename _RAIter> typedef std::pair<_RAIter, _RAIter> __gnu_parallel::__QSBThreadLocal<_RAIter >::__Piece`

Continuous part of the sequence, described by an iterator pair.

Definition at line 72 of file `balanced_quicksort.h`.

5.161.3 Constructor & Destructor Documentation

5.161.3.1 `template<typename _RAIter> __gnu_parallel::__QSBThreadLocal<_RAIter >::__QSBThreadLocal (int __queue_size) [inline]`

Constructor.

Parameters

<code>__queue_size</code>	size of the work-stealing queue.
---------------------------	----------------------------------

Definition at line 91 of file `balanced_quicksort.h`.

5.161.4 Member Data Documentation

5.161.4.1 `template<typename _RAIter> volatile _DifferenceType* __gnu_parallel::__QSBThreadLocal<_RAIter>::__M_elements_leftover`

Pointer to a counter of elements left over to sort.

Definition at line 84 of file `balanced_quicksort.h`.

Referenced by `__gnu_parallel::__parallel_sort_qsb()`, `__gnu_parallel::__qsb_conquer()`, and `__gnu_parallel::__qsb_local_sort_with_helping()`.

5.161.4.2 `template<typename _RAIter> _Piece __gnu_parallel::__QSBThreadLocal<_RAIter>::__M_global`

The complete sequence to sort.

Definition at line 87 of file `balanced_quicksort.h`.

5.161.4.3 `template<typename _RAIter> _Piece __gnu_parallel::__QSBThreadLocal<_RAIter>::__M_initial`

Initial piece to work on.

Definition at line 75 of file `balanced_quicksort.h`.

Referenced by `__gnu_parallel::__qsb_conquer()`, and `__gnu_parallel::__qsb_local_sort_with_helping()`.

5.161.4.4 `template<typename _RAIter> _RestrictedBoundedConcurrentQueue<_Piece> __gnu_parallel::__QSBThreadLocal<_RAIter>::__M_leftover_parts`

Work-stealing queue.

Definition at line 78 of file `balanced_quicksort.h`.

Referenced by `__gnu_parallel::__parallel_sort_qsb()`, and `__gnu_parallel::__qsb_local_sort_with_helping()`.

5.161.4.5 `template<typename _RAIter> _ThreadIndex __gnu_parallel::__QSBThreadLocal<_RAIter>::__M_num_threads`

Number of threads involved in this algorithm.

Definition at line 81 of file `balanced_quicksort.h`.

Referenced by `__gnu_parallel::__qsb_local_sort_with_helping()`.

The documentation for this struct was generated from the following file:

- [balanced_quicksort.h](#)

5.162 `__gnu_parallel::_RandomNumber` Class Reference

Public Member Functions

- [`_RandomNumber`](#) ()
- [`_RandomNumber`](#) (uint32_t __seed, uint64_t _M_supremum=0x100000000ULL)
- unsigned long [`__genrand_bits`](#) (int __bits)
- uint32_t [`operator\(\)`](#) ()
- uint32_t [`operator\(\)`](#) (uint64_t local_supremum)

5.162.1 Detailed Description

Random number generator, based on the Mersenne twister.

Definition at line 42 of file `random_number.h`.

5.162.2 Constructor & Destructor Documentation

5.162.2.1 `__gnu_parallel::_RandomNumber::_RandomNumber ()` `[inline]`

Default constructor. Seed with 0.

Definition at line 74 of file `random_number.h`.

5.162.2.2 `__gnu_parallel::_RandomNumber::_RandomNumber (uint32_t __seed, uint64_t _M_supremum = 0x100000000ULL)` `[inline]`

Constructor.

Parameters

<code>__seed</code>	Random __seed.
<code>_M_supremum</code>	Generate integer random numbers in the interval [0, _M_supremum).

Definition at line 85 of file `random_number.h`.

5.162.3 Member Function Documentation

5.162.3.1 `unsigned long __gnu_parallel::_RandomNumber::__genrand_bits (int __bits)` `[inline]`

Generate a number of random bits, run-time parameter.

Parameters

<code>__bits</code>	Number of bits to generate.
---------------------	-----------------------------

Definition at line 109 of file `random_number.h`.

5.162.3.2 `uint32_t __gnu_parallel::RandomNumber::operator() () [inline]`

Generate unsigned random 32-bit integer.

Definition at line 94 of file `random_number.h`.

5.162.3.3 `uint32_t __gnu_parallel::RandomNumber::operator() (uint64_t local_supremum) [inline]`

Generate unsigned random 32-bit integer in the interval `[0,local_supremum)`.

Definition at line 100 of file `random_number.h`.

The documentation for this class was generated from the following file:

- [random_number.h](#)

5.163 `__gnu_parallel::_RestrictedBoundedConcurrentQueue<_Tp>` Class Template Reference

Public Member Functions

- [_RestrictedBoundedConcurrentQueue](#) ([_SequenceIndex](#) __max_size)
- [~_RestrictedBoundedConcurrentQueue](#) ()
- bool [pop_back](#) ([_Tp](#) &__t)
- bool [pop_front](#) ([_Tp](#) &__t)
- void [push_front](#) (const [_Tp](#) &__t)

5.163.1 Detailed Description

```
template<typename _Tp>
class __gnu_parallel::_RestrictedBoundedConcurrentQueue<_Tp>
```

Double-ended queue of bounded size, allowing lock-free atomic access. `push_front()` and `pop_front()` must not be called concurrently to each other, while `pop_back()` can be called concurrently at all times. `empty()`, `size()`, and `top()` are intentionally not provided. Calling them would not make sense in a concurrent setting.

Parameters

<code>_Tp</code>	Contained element type.
------------------	-------------------------

Definition at line 52 of file queue.h.

5.163.2 Constructor & Destructor Documentation

5.163.2.1 `template<typename _Tp> __gnu_parallel::_RestrictedBoundedConcurrentQueue< _Tp
>::_RestrictedBoundedConcurrentQueue (_SequenceIndex __max_size) [inline]`

Constructor. Not to be called concurrent, of course.

Parameters

<code>__max_size</code>	Maximal number of elements to be contained.
-------------------------	---

Definition at line 68 of file queue.h.

5.163.2.2 `template<typename _Tp> __gnu_parallel::_RestrictedBoundedConcurrentQueue< _Tp
>::~~_RestrictedBoundedConcurrentQueue () [inline]`

Destructor. Not to be called concurrent, of course.

Definition at line 77 of file queue.h.

5.163.3 Member Function Documentation

5.163.3.1 `template<typename _Tp> bool __gnu_parallel::_RestrictedBoundedConcurrentQueue< _Tp >::pop_back (_Tp & __t) [inline]`

Pops one element from the queue at the front end. Must not be called concurrently with pop_front().

Definition at line 127 of file queue.h.

5.163.3.2 `template<typename _Tp> bool __gnu_parallel::_RestrictedBoundedConcurrentQueue< _Tp >::pop_front (_Tp & __t) [inline]`

Pops one element from the queue at the front end. Must not be called concurrently with pop_front().

Definition at line 100 of file queue.h.

5.163.3.3 `template<typename _Tp> void __gnu_parallel::_RestrictedBoundedConcurrentQueue< _Tp >::push_front (const _Tp & __t) [inline]`

Pushes one element into the queue at the front end. Must not be called concurrently with pop_front().

Definition at line 83 of file queue.h.

The documentation for this class was generated from the following file:

- [queue.h](#)

5.164 `__gnu_parallel::_SamplingSorter< __stable, _RAIter, _StrictWeakOrdering >` Struct Template Reference

Public Member Functions

- void **operator()** (_RAIter __first, _RAIter __last, _StrictWeakOrdering __comp)

5.164.1 Detailed Description

```
template<bool __stable, class _RAIter, class _StrictWeakOrdering>
struct __gnu_parallel::_SamplingSorter< __stable, _RAIter, _StrictWeakOrdering >
```

Stable sorting functor.

Used to reduce code instantiation in `multiway_merge_sampling_splitting`.

Definition at line 1007 of file `multiway_merge.h`.

The documentation for this struct was generated from the following file:

- [multiway_merge.h](#)

5.165 `__gnu_parallel::_SamplingSorter< false, _RAIter, _StrictWeakOrdering >` Struct Template Reference

Public Member Functions

- void **operator()** (_RAIter __first, _RAIter __last, _StrictWeakOrdering __comp)

5.165.1 Detailed Description

```
template<class _RAIter, class _StrictWeakOrdering>
struct __gnu_parallel::_SamplingSorter< false, _RAIter, _StrictWeakOrdering >
```

Non-`__stable` sorting functor.

Used to reduce code instantiation in `multiway_merge_sampling_splitting`.

Definition at line 1020 of file `multiway_merge.h`.

The documentation for this struct was generated from the following file:

- [multiway_merge.h](#)

5.166 `__gnu_parallel::_Settings` Struct Reference

Static Public Member Functions

- static const `_Settings` & `get` () throw ()
- static void `set` (`_Settings` &) throw ()

Public Attributes

- `_SequenceIndex` `accumulate_minimal_n`
- unsigned int `adjacent_difference_minimal_n`
- `_AlgorithmStrategy` `algorithm_strategy`
- unsigned int `cache_line_size`
- `_SequenceIndex` `count_minimal_n`
- `_SequenceIndex` `fill_minimal_n`
- `_FindAlgorithm` `find_algorithm`
- double `find_increasing_factor`
- `_SequenceIndex` `find_initial_block_size`
- `_SequenceIndex` `find_maximum_block_size`
- float `find_scale_factor`
- `_SequenceIndex` `find_sequential_search_size`
- `_SequenceIndex` `for_each_minimal_n`
- `_SequenceIndex` `generate_minimal_n`
- unsigned long long `L1_cache_size`
- unsigned long long `L2_cache_size`
- `_SequenceIndex` `max_element_minimal_n`
- `_SequenceIndex` `merge_minimal_n`
- unsigned int `merge_oversampling`
- `_SplittingAlgorithm` `merge_splitting`
- `_SequenceIndex` `min_element_minimal_n`
- `_MultiwayMergeAlgorithm` `multiway_merge_algorithm`
- int `multiway_merge_minimal_k`
- `_SequenceIndex` `multiway_merge_minimal_n`
- unsigned int `multiway_merge_oversampling`
- `_SplittingAlgorithm` `multiway_merge_splitting`
- `_SequenceIndex` `nth_element_minimal_n`
- `_SequenceIndex` `partial_sort_minimal_n`
- `_PartialSumAlgorithm` `partial_sum_algorithm`
- float `partial_sum_dilation`
- unsigned int `partial_sum_minimal_n`
- double `partition_chunk_share`
- `_SequenceIndex` `partition_chunk_size`
- `_SequenceIndex` `partition_minimal_n`
- `_SequenceIndex` `qsb_steals`
- unsigned int `random_shuffle_minimal_n`
- `_SequenceIndex` `replace_minimal_n`
- `_SequenceIndex` `search_minimal_n`
- `_SequenceIndex` `set_difference_minimal_n`
- `_SequenceIndex` `set_intersection_minimal_n`
- `_SequenceIndex` `set_symmetric_difference_minimal_n`

- [_SequenceIndex](#) `set_union_minimal_n`
- [_SortAlgorithm](#) `sort_algorithm`
- [_SequenceIndex](#) `sort_minimal_n`
- unsigned int `sort_mwms_oversampling`
- unsigned int `sort_qs_num_samples_preset`
- [_SequenceIndex](#) `sort_qsb_base_case_maximal_n`
- [_SplittingAlgorithm](#) `sort_splitting`
- unsigned int `TLB_size`
- [_SequenceIndex](#) `transform_minimal_n`
- [_SequenceIndex](#) `unique_copy_minimal_n`
- [_SequenceIndex](#) `workstealing_chunk_size`

5.166.1 Detailed Description

class `_Settings` Run-time settings for the parallel mode including all tunable parameters.

Definition at line 123 of file `settings.h`.

5.166.2 Member Function Documentation

5.166.2.1 `static const _Settings& __gnu_parallel::Settings::get() throw()` `[static]`

Get the global settings.

Referenced by `__gnu_parallel::__find_template()`, `__gnu_parallel::__for_each_template_random_access_workstealing()`, `__gnu_parallel::__parallel_nth_element()`, `__gnu_parallel::__parallel_partial_sum()`, `__gnu_parallel::__parallel_partial_sum_linear()`, `__gnu_parallel::__parallel_partition()`, `__gnu_parallel::__parallel_random_shuffle_drs()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort_qs_conquer()`, `__gnu_parallel::__qsb_local_sort_with_helping()`, `__gnu_parallel::__sequential_random_shuffle()`, `__gnu_parallel::multiway_merge()`, `__gnu_parallel::multiway_merge_sampling_splitting()`, `__gnu_parallel::multiway_merge_sentinels()`, `__gnu_parallel::parallel_multiway_merge()`, `__gnu_parallel::parallel_sort_mwms()`, and `__gnu_parallel::parallel_sort_mwms_pu()`.

5.166.2.2 `static void __gnu_parallel::Settings::set(_Settings &) throw()` `[static]`

Set the global settings.

5.166.3 Member Data Documentation

5.166.3.1 `_SequenceIndex __gnu_parallel::Settings::accumulate_minimal_n`

Minimal input size for accumulate.

Definition at line 139 of file `settings.h`.

5.166.3.2 `unsigned int __gnu_parallel::_Settings::adjacent_difference_minimal_n`

Minimal input size for `adjacent_difference`.

Definition at line 142 of file `settings.h`.

5.166.3.3 `unsigned int __gnu_parallel::_Settings::cache_line_size`

Overestimation of cache line size. Used to avoid false sharing, i.e. elements of different threads are at least this amount apart.

Definition at line 265 of file `settings.h`.

Referenced by `__gnu_parallel::_for_each_template_random_access_workstealing()`.

5.166.3.4 `_SequenceIndex __gnu_parallel::_Settings::count_minimal_n`

Minimal input size for `count` and `count_if`.

Definition at line 145 of file `settings.h`.

5.166.3.5 `_SequenceIndex __gnu_parallel::_Settings::fill_minimal_n`

Minimal input size for `fill`.

Definition at line 148 of file `settings.h`.

5.166.3.6 `double __gnu_parallel::_Settings::find_increasing_factor`

Block size increase factor for `find`.

Definition at line 151 of file `settings.h`.

5.166.3.7 `_SequenceIndex __gnu_parallel::_Settings::find_initial_block_size`

Initial block size for `find`.

Definition at line 154 of file `settings.h`.

Referenced by `__gnu_parallel::_find_template()`.

5.166.3.8 `_SequenceIndex __gnu_parallel::_Settings::find_maximum_block_size`

Maximal block size for `find`.

Definition at line 157 of file `settings.h`.

5.166.3.9 `float __gnu_parallel::_Settings::find_scale_factor`

Block size scale-down factor with respect to current position.

Definition at line 276 of file settings.h.

Referenced by `__gnu_parallel::_find_template()`.

5.166.3.10 `_SequenceIndex __gnu_parallel::_Settings::find_sequential_search_size`

Start with looking for this many elements sequentially, for find.

Definition at line 160 of file settings.h.

Referenced by `__gnu_parallel::_find_template()`.

5.166.3.11 `_SequenceIndex __gnu_parallel::_Settings::for_each_minimal_n`

Minimal input size for `for_each`.

Definition at line 163 of file settings.h.

5.166.3.12 `_SequenceIndex __gnu_parallel::_Settings::generate_minimal_n`

Minimal input size for `generate`.

Definition at line 166 of file settings.h.

5.166.3.13 `unsigned long long __gnu_parallel::_Settings::L1_cache_size`

size of the L1 cache in bytes (underestimation).

Definition at line 254 of file settings.h.

5.166.3.14 `unsigned long long __gnu_parallel::_Settings::L2_cache_size`

size of the L2 cache in bytes (underestimation).

Definition at line 257 of file settings.h.

Referenced by `__gnu_parallel::_parallel_random_shuffle_drs()`, and `__gnu_parallel::_sequential_random_shuffle()`.

5.166.3.15 `_SequenceIndex __gnu_parallel::_Settings::max_element_minimal_n`

Minimal input size for `max_element`.

Definition at line 169 of file settings.h.

5.166.3.16 `_SequenceIndex __gnu_parallel::_Settings::merge_minimal_n`

Minimal input size for merge.

Definition at line 172 of file settings.h.

5.166.3.17 `unsigned int __gnu_parallel::_Settings::merge_oversampling`

Oversampling factor for merge.

Definition at line 175 of file settings.h.

Referenced by `__gnu_parallel::multiway_merge_sampling_splitting()`, and `__gnu_parallel::parallel_multiway_merge()`.

5.166.3.18 `_SequenceIndex __gnu_parallel::_Settings::min_element_minimal_n`

Minimal input size for `min_element`.

Definition at line 178 of file settings.h.

5.166.3.19 `int __gnu_parallel::_Settings::multiway_merge_minimal_k`

Oversampling factor for `multiway_merge`.

Definition at line 184 of file settings.h.

5.166.3.20 `_SequenceIndex __gnu_parallel::_Settings::multiway_merge_minimal_n`

Minimal input size for `multiway_merge`.

Definition at line 181 of file settings.h.

5.166.3.21 `unsigned int __gnu_parallel::_Settings::multiway_merge_oversampling`

Oversampling factor for `multiway_merge`.

Definition at line 187 of file settings.h.

5.166.3.22 `_SequenceIndex __gnu_parallel::_Settings::nth_element_minimal_n`

Minimal input size for `nth_element`.

Definition at line 190 of file settings.h.

Referenced by `__gnu_parallel::__parallel_nth_element()`.

5.166.3.23 `_SequenceIndex __gnu_parallel::_Settings::partial_sort_minimal_n`

Minimal input size for `partial_sort`.

Definition at line 203 of file settings.h.

5.166.3.24 `float __gnu_parallel::_Settings::partial_sum_dilation`

Ratio for `partial_sum`. Assume "sum and write result" to be this factor slower than just "sum".

Definition at line 207 of file `settings.h`.

Referenced by `__gnu_parallel::_parallel_partial_sum_linear()`.

5.166.3.25 `unsigned int __gnu_parallel::_Settings::partial_sum_minimal_n`

Minimal input size for `partial_sum`.

Definition at line 210 of file `settings.h`.

5.166.3.26 `double __gnu_parallel::_Settings::partition_chunk_share`

Chunk size for partition, relative to input size. If > 0.0 , this value overrides `partition_chunk_size`.

Definition at line 197 of file `settings.h`.

Referenced by `__gnu_parallel::_parallel_partition()`.

5.166.3.27 `_SequenceIndex __gnu_parallel::_Settings::partition_chunk_size`

Chunk size for partition.

Definition at line 193 of file `settings.h`.

Referenced by `__gnu_parallel::_parallel_partition()`.

5.166.3.28 `_SequenceIndex __gnu_parallel::_Settings::partition_minimal_n`

Minimal input size for partition.

Definition at line 200 of file `settings.h`.

Referenced by `__gnu_parallel::_parallel_nth_element()`.

5.166.3.29 `_SequenceIndex __gnu_parallel::_Settings::qsb_steals`

The number of stolen ranges in load-balanced quicksort.

Definition at line 270 of file `settings.h`.

5.166.3.30 `unsigned int __gnu_parallel::_Settings::random_shuffle_minimal_n`

Minimal input size for `random_shuffle`.

Definition at line 213 of file `settings.h`.

5.166.3.31 `__SequenceIndex __gnu_parallel::_Settings::replace_minimal_n`

Minimal input size for `replace` and `replace_if`.

Definition at line 216 of file `settings.h`.

5.166.3.32 `__SequenceIndex __gnu_parallel::_Settings::search_minimal_n`

Minimal input size for `search` and `search_n`.

Definition at line 273 of file `settings.h`.

5.166.3.33 `__SequenceIndex __gnu_parallel::_Settings::set_difference_minimal_n`

Minimal input size for `set_difference`.

Definition at line 219 of file `settings.h`.

5.166.3.34 `__SequenceIndex __gnu_parallel::_Settings::set_intersection_minimal_n`

Minimal input size for `set_intersection`.

Definition at line 222 of file `settings.h`.

5.166.3.35 `__SequenceIndex __gnu_parallel::_Settings::set_symmetric_difference_minimal_n`

Minimal input size for `set_symmetric_difference`.

Definition at line 225 of file `settings.h`.

5.166.3.36 `__SequenceIndex __gnu_parallel::_Settings::set_union_minimal_n`

Minimal input size for `set_union`.

Definition at line 228 of file `settings.h`.

5.166.3.37 `__SequenceIndex __gnu_parallel::_Settings::sort_minimal_n`

Minimal input size for parallel sorting.

Definition at line 231 of file `settings.h`.

5.166.3.38 `unsigned int __gnu_parallel::_Settings::sort_mwms_oversampling`

Oversampling factor for parallel `std::sort` (MWMS).

Definition at line 234 of file `settings.h`.

Referenced by `__gnu_parallel::parallel_sort_mwms()`, and `__gnu_parallel::parallel_sort_mwms_pu()`.

5.166.3.39 `unsigned int __gnu_parallel::_Settings::sort_qs_num_samples_preset`

Such many samples to take to find a good pivot (quicksort).

Definition at line 237 of file settings.h.

5.166.3.40 `_SequenceIndex __gnu_parallel::_Settings::sort_qsb_base_case_maximal_n`

Maximal subsequence `__length` to switch to unbalanced `__base` case. Applies to `std::sort` with dynamically load-balanced quicksort.

Definition at line 241 of file settings.h.

Referenced by `__gnu_parallel::__qsb_local_sort_with_helping()`.

5.166.3.41 `unsigned int __gnu_parallel::_Settings::TLB_size`

size of the Translation Lookaside Buffer (underestimation).

Definition at line 260 of file settings.h.

Referenced by `__gnu_parallel::__parallel_random_shuffle_drs()`, and `__gnu_parallel::__sequential_random_shuffle()`.

5.166.3.42 `_SequenceIndex __gnu_parallel::_Settings::transform_minimal_n`

Minimal input size for parallel `std::transform`.

Definition at line 244 of file settings.h.

5.166.3.43 `_SequenceIndex __gnu_parallel::_Settings::unique_copy_minimal_n`

Minimal input size for `unique_copy`.

Definition at line 247 of file settings.h.

The documentation for this struct was generated from the following file:

- [settings.h](#)

5.167 `__gnu_parallel::_SplitConsistently< __exact, _RAIter, _Compare, _SortingPlacesIterator > Struct` Template Reference

5.167.1 Detailed Description

```
template<bool __exact, typename _RAIter, typename _Compare, typename _SortingPlacesIterator>
struct __gnu_parallel::_SplitConsistently< __exact, _RAIter, _Compare, _SortingPlacesIterator >
```

Split consistently.

Definition at line 122 of file multiway_mergesort.h.

The documentation for this struct was generated from the following file:

- [multiway_mergesort.h](#)

5.168 `__gnu_parallel::_SplitConsistently< false, _RAIter, _Compare, _SortingPlacesIter >` Struct Template Reference

Public Member Functions

- void **operator()** (const [_ThreadIndex](#) __iam, [_PMWSSortingData](#)< _RAIter > *__sd, _Compare &__comp, const typename std::iterator_traits< _RAIter >::difference_type __num_samples) const

5.168.1 Detailed Description

```
template<typename _RAIter, typename _Compare, typename _SortingPlacesIter>
struct __gnu_parallel::_SplitConsistently< false, _RAIter, _Compare, _SortingPlacesIter >
```

Split by sampling.

Definition at line 187 of file `multiway_mergesort.h`.

The documentation for this struct was generated from the following file:

- [multiway_mergesort.h](#)

5.169 `__gnu_parallel::_SplitConsistently< true, _RAIter, _Compare, _SortingPlacesIter >` Struct Template Reference

Public Member Functions

- void **operator()** (const [_ThreadIndex](#) __iam, [_PMWSSortingData](#)< _RAIter > *__sd, _Compare &__comp, const typename std::iterator_traits< _RAIter >::difference_type __num_samples) const

5.169.1 Detailed Description

```
template<typename _RAIter, typename _Compare, typename _SortingPlacesIter>
struct __gnu_parallel::_SplitConsistently< true, _RAIter, _Compare, _SortingPlacesIter >
```

Split by exact splitting.

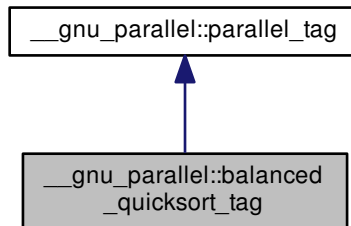
Definition at line 128 of file `multiway_mergesort.h`.

The documentation for this struct was generated from the following file:

- [multiway_mergesort.h](#)

5.170 `__gnu_parallel::balanced_quicksort_tag` Struct Reference

Inheritance diagram for `__gnu_parallel::balanced_quicksort_tag`:



Public Member Functions

- **`balanced_quicksort_tag`** (`_ThreadIndex` `__num_threads`)
- `_ThreadIndex` `__get_num_threads` ()
- void `set_num_threads` (`_ThreadIndex` `__num_threads`)

5.170.1 Detailed Description

Forces parallel sorting using balanced quicksort at compile time.

Definition at line 164 of file `tags.h`.

5.170.2 Member Function Documentation

5.170.2.1 `_ThreadIndex __gnu_parallel::parallel_tag::__get_num_threads` () `[inline]`, `[inherited]`

Find out desired number of threads.

Returns

Desired number of threads.

Definition at line 63 of file `tags.h`.

Referenced by `__gnu_parallel::__parallel_sort`(), `__gnu_parallel::multiway_merge`(), and `__gnu_parallel::multiway_merge_sentinels`().

5.170.2.2 void `__gnu_parallel::parallel_tag::set_num_threads` (`_ThreadIndex` `__num_threads`) `[inline]`, `[inherited]`

Set the desired number of threads.

Parameters

<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

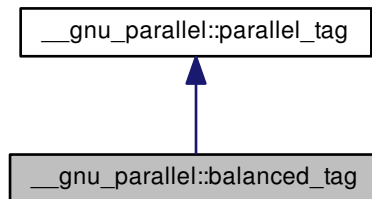
Definition at line 73 of file tags.h.

The documentation for this struct was generated from the following file:

- [tags.h](#)

5.171 __gnu_parallel::balanced_tag Struct Reference

Inheritance diagram for `__gnu_parallel::balanced_tag`:

**Public Member Functions**

- [_ThreadIndex __get_num_threads\(\)](#)
- void [set_num_threads\(_ThreadIndex __num_threads\)](#)

5.171.1 Detailed Description

Recommends parallel execution using dynamic load-balancing at compile time.

Definition at line 88 of file tags.h.

5.171.2 Member Function Documentation**5.171.2.1 _ThreadIndex __gnu_parallel::parallel_tag::__get_num_threads() [inline],[inherited]**

Find out desired number of threads.

Returns

Desired number of threads.

Definition at line 63 of file tags.h.

Referenced by `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::multiway_merge()`, and `__gnu_parallel::multiway_merge_sentinels()`.

5.171.2.2 `void __gnu_parallel::parallel_tag::set_num_threads (_ThreadIndex __num_threads) [inline], [inherited]`

Set the desired number of threads.

Parameters

<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

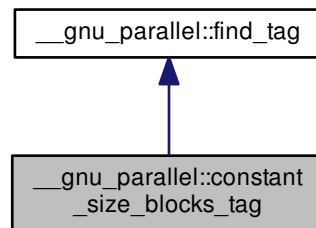
Definition at line 73 of file `tags.h`.

The documentation for this struct was generated from the following file:

- [tags.h](#)

5.172 `__gnu_parallel::constant_size_blocks_tag` Struct Reference

Inheritance diagram for `__gnu_parallel::constant_size_blocks_tag`:



5.172.1 Detailed Description

Selects the constant block size variant for `std::find()`.

See also

`_GLIBCXX_FIND_CONSTANT_SIZE_BLOCKS`

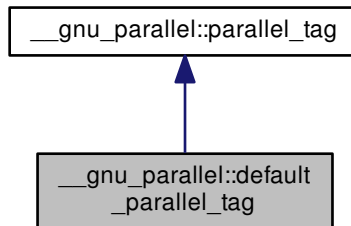
Definition at line 178 of file `tags.h`.

The documentation for this struct was generated from the following file:

- [tags.h](#)

5.173 `__gnu_parallel::default_parallel_tag` Struct Reference

Inheritance diagram for `__gnu_parallel::default_parallel_tag`:



Public Member Functions

- **`default_parallel_tag`** (`_ThreadIndex` `__num_threads`)
- `_ThreadIndex` `__get_num_threads` ()
- void `set_num_threads` (`_ThreadIndex` `__num_threads`)

5.173.1 Detailed Description

Recommends parallel execution using the default parallel algorithm.

Definition at line 79 of file `tags.h`.

5.173.2 Member Function Documentation

5.173.2.1 `_ThreadIndex __gnu_parallel::parallel_tag::__get_num_threads` () `[inline]`, `[inherited]`

Find out desired number of threads.

Returns

Desired number of threads.

Definition at line 63 of file `tags.h`.

Referenced by `__gnu_parallel::__parallel_sort`(), `__gnu_parallel::multiway_merge`(), and `__gnu_parallel::multiway_merge_sentinels`().

5.173.2.2 void `__gnu_parallel::parallel_tag::set_num_threads` (`_ThreadIndex` `__num_threads`) `[inline]`, `[inherited]`

Set the desired number of threads.

Parameters

<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

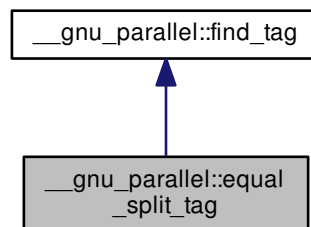
Definition at line 73 of file tags.h.

The documentation for this struct was generated from the following file:

- [tags.h](#)

5.174 __gnu_parallel::equal_split_tag Struct Reference

Inheritance diagram for __gnu_parallel::equal_split_tag:



5.174.1 Detailed Description

Selects the equal splitting variant for `std::find()`.

See also

`_GLIBCXX_FIND_EQUAL_SPLIT`

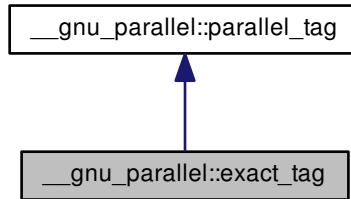
Definition at line 182 of file tags.h.

The documentation for this struct was generated from the following file:

- [tags.h](#)

5.175 `__gnu_parallel::exact_tag` Struct Reference

Inheritance diagram for `__gnu_parallel::exact_tag`:



Public Member Functions

- **`exact_tag`** (`_ThreadIndex` `__num_threads`)
- `_ThreadIndex` `__get_num_threads` ()
- void `set_num_threads` (`_ThreadIndex` `__num_threads`)

5.175.1 Detailed Description

Forces parallel merging with exact splitting, at compile time.

Definition at line 109 of file tags.h.

5.175.2 Member Function Documentation

5.175.2.1 `_ThreadIndex` `__gnu_parallel::parallel_tag::__get_num_threads` () `[inline]`, `[inherited]`

Find out desired number of threads.

Returns

Desired number of threads.

Definition at line 63 of file tags.h.

Referenced by `__gnu_parallel::__parallel_sort`(), `__gnu_parallel::multiway_merge`(), and `__gnu_parallel::multiway_merge_sentinels`().

5.175.2.2 void `__gnu_parallel::parallel_tag::set_num_threads` (`_ThreadIndex` `__num_threads`) `[inline]`, `[inherited]`

Set the desired number of threads.

Parameters

<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

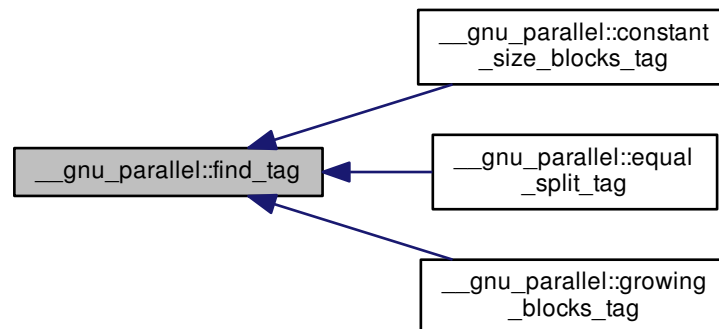
Definition at line 73 of file tags.h.

The documentation for this struct was generated from the following file:

- [tags.h](#)

5.176 __gnu_parallel::find_tag Struct Reference

Inheritance diagram for __gnu_parallel::find_tag:



5.176.1 Detailed Description

Base class for for `std::find()` variants.

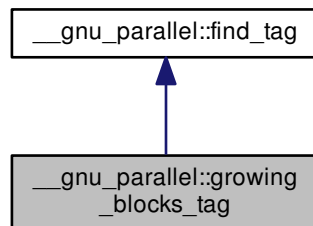
Definition at line 104 of file tags.h.

The documentation for this struct was generated from the following file:

- [tags.h](#)

5.177 `__gnu_parallel::growing_blocks_tag` Struct Reference

Inheritance diagram for `__gnu_parallel::growing_blocks_tag`:



5.177.1 Detailed Description

Selects the growing block size variant for `std::find()`.

See also

`_GLIBCXX_FIND_GROWING_BLOCKS`

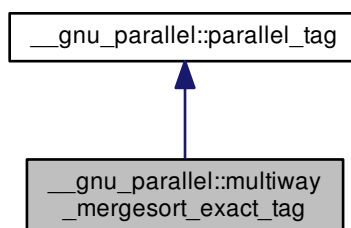
Definition at line 174 of file `tags.h`.

The documentation for this struct was generated from the following file:

- [tags.h](#)

5.178 `__gnu_parallel::multiway_mergesort_exact_tag` Struct Reference

Inheritance diagram for `__gnu_parallel::multiway_mergesort_exact_tag`:



Public Member Functions

- **`multiway_mergesort_exact_tag`** ([_ThreadIndex](#) __num_threads)
- [_ThreadIndex](#) `__get_num_threads` ()
- void `set_num_threads` ([_ThreadIndex](#) __num_threads)

5.178.1 Detailed Description

Forces parallel sorting using multiway mergesort with exact splitting at compile time.

Definition at line 137 of file tags.h.

5.178.2 Member Function Documentation

5.178.2.1 [_ThreadIndex](#) `__gnu_parallel::parallel_tag::__get_num_threads` () [[inline](#)], [[inherited](#)]

Find out desired number of threads.

Returns

Desired number of threads.

Definition at line 63 of file tags.h.

Referenced by `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::multiway_merge()`, and `__gnu_parallel::multiway_merge_sentinels()`.

5.178.2.2 void `__gnu_parallel::parallel_tag::set_num_threads` ([_ThreadIndex](#) __num_threads) [[inline](#)], [[inherited](#)]

Set the desired number of threads.

Parameters

__num_threads	Desired number of threads.
-------------------------------	----------------------------

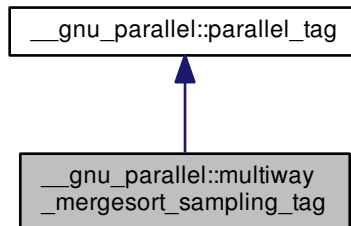
Definition at line 73 of file tags.h.

The documentation for this struct was generated from the following file:

- [tags.h](#)

5.179 __gnu_parallel::multiway_mergesort_sampling_tag Struct Reference

Inheritance diagram for __gnu_parallel::multiway_mergesort_sampling_tag:



Public Member Functions

- **multiway_mergesort_sampling_tag** ([_ThreadIndex](#) __num_threads)
- [_ThreadIndex](#) **__get_num_threads** ()
- void **set_num_threads** ([_ThreadIndex](#) __num_threads)

5.179.1 Detailed Description

Forces parallel sorting using multiway mergesort with splitting by sampling at compile time.

Definition at line 146 of file tags.h.

5.179.2 Member Function Documentation

5.179.2.1 [_ThreadIndex](#) __gnu_parallel::parallel_tag::__get_num_threads () [\[inline\]](#), [\[inherited\]](#)

Find out desired number of threads.

Returns

Desired number of threads.

Definition at line 63 of file tags.h.

Referenced by `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::multiway_merge()`, and `__gnu_parallel::multiway_merge_sentinels()`.

5.179.2.2 void __gnu_parallel::parallel_tag::set_num_threads ([_ThreadIndex](#) __num_threads) [\[inline\]](#), [\[inherited\]](#)

Set the desired number of threads.

Parameters

<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

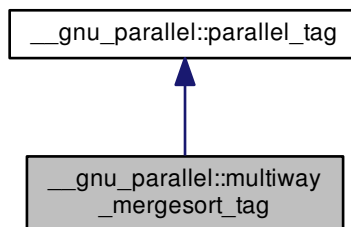
Definition at line 73 of file tags.h.

The documentation for this struct was generated from the following file:

- [tags.h](#)

5.180 `__gnu_parallel::multiway_mergesort_tag` Struct Reference

Inheritance diagram for `__gnu_parallel::multiway_mergesort_tag`:



Public Member Functions

- **`multiway_mergesort_tag`** ([_ThreadIndex](#) __num_threads)
- [_ThreadIndex](#) `__get_num_threads` ()
- void `set_num_threads` ([_ThreadIndex](#) __num_threads)

5.180.1 Detailed Description

Forces parallel sorting using multiway mergesort at compile time.

Definition at line 128 of file tags.h.

5.180.2 Member Function Documentation

5.180.2.1 `_ThreadIndex __gnu_parallel::parallel_tag::get_num_threads ()` `[inline]`, `[inherited]`

Find out desired number of threads.

Returns

Desired number of threads.

Definition at line 63 of file tags.h.

Referenced by `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::multiway_merge()`, and `__gnu_parallel::multiway_merge_sentinels()`.

5.180.2.2 `void __gnu_parallel::parallel_tag::set_num_threads (_ThreadIndex __num_threads)` `[inline]`, `[inherited]`

Set the desired number of threads.

Parameters

<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

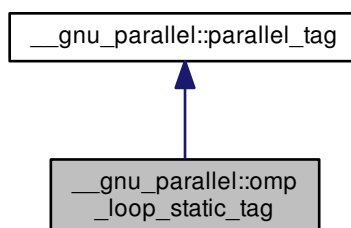
Definition at line 73 of file tags.h.

The documentation for this struct was generated from the following file:

- [tags.h](#)

5.181 `__gnu_parallel::omp_loop_static_tag` Struct Reference

Inheritance diagram for `__gnu_parallel::omp_loop_static_tag`:



Public Member Functions

- [_ThreadIndex __get_num_threads\(\)](#)
- void [set_num_threads\(_ThreadIndex __num_threads\)](#)

5.181.1 Detailed Description

Recommends parallel execution using OpenMP static load-balancing at compile time.

Definition at line 100 of file `tags.h`.

5.181.2 Member Function Documentation

5.181.2.1 `_ThreadIndex __gnu_parallel::parallel_tag::__get_num_threads()` `[inline]`, `[inherited]`

Find out desired number of threads.

Returns

Desired number of threads.

Definition at line 63 of file `tags.h`.

Referenced by `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::multiway_merge()`, and `__gnu_parallel::multiway_merge_sentinels()`.

5.181.2.2 `void __gnu_parallel::parallel_tag::set_num_threads(_ThreadIndex __num_threads)` `[inline]`, `[inherited]`

Set the desired number of threads.

Parameters

<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

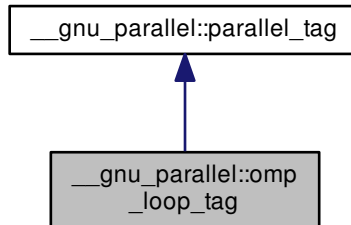
Definition at line 73 of file `tags.h`.

The documentation for this struct was generated from the following file:

- [tags.h](#)

5.182 __gnu_parallel::omp_loop_tag Struct Reference

Inheritance diagram for __gnu_parallel::omp_loop_tag:



Public Member Functions

- [_ThreadIndex __get_num_threads\(\)](#)
- void [set_num_threads\(_ThreadIndex __num_threads\)](#)

5.182.1 Detailed Description

Recommends parallel execution using OpenMP dynamic load-balancing at compile time.

Definition at line 96 of file tags.h.

5.182.2 Member Function Documentation

5.182.2.1 _ThreadIndex __gnu_parallel::parallel_tag::__get_num_threads() [inline], [inherited]

Find out desired number of threads.

Returns

Desired number of threads.

Definition at line 63 of file tags.h.

Referenced by `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::multiway_merge()`, and `__gnu_parallel::multiway_merge_sentinels()`.

5.182.2.2 void __gnu_parallel::parallel_tag::set_num_threads(_ThreadIndex __num_threads) [inline], [inherited]

Set the desired number of threads.

Parameters

<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

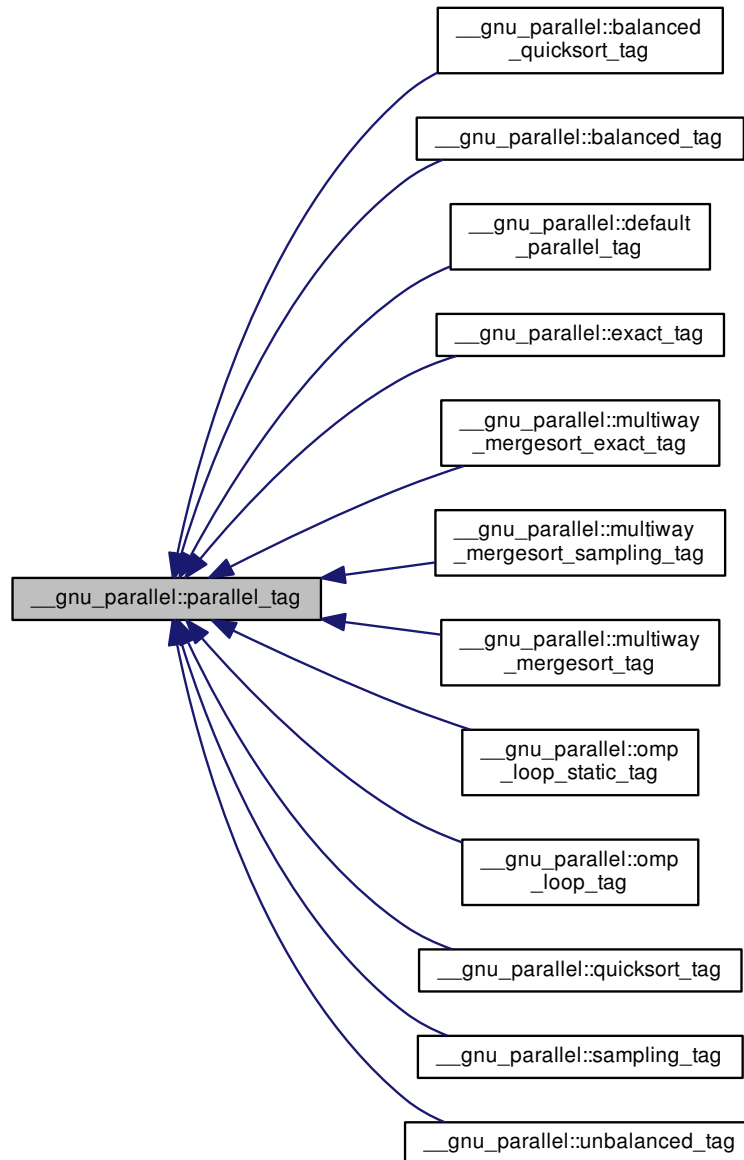
Definition at line 73 of file `tags.h`.

The documentation for this struct was generated from the following file:

- [tags.h](#)

5.183 __gnu_parallel::parallel_tag Struct Reference

Inheritance diagram for __gnu_parallel::parallel_tag:



Public Member Functions

- [parallel_tag](#) ()
- [parallel_tag](#) (_ThreadIndex __num_threads)
- [_ThreadIndex __get_num_threads](#) ()
- void [set_num_threads](#) (_ThreadIndex __num_threads)

5.183.1 Detailed Description

Recommends parallel execution at compile time, optionally using a user-specified number of threads.

Definition at line 46 of file tags.h.

5.183.2 Constructor & Destructor Documentation

5.183.2.1 `__gnu_parallel::parallel_tag::parallel_tag ()` `[inline]`

Default constructor. Use default number of threads.

Definition at line 53 of file tags.h.

5.183.2.2 `__gnu_parallel::parallel_tag::parallel_tag (_ThreadIndex __num_threads)` `[inline]`

Default constructor. Recommend number of threads to use.

Parameters

<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

Definition at line 58 of file tags.h.

5.183.3 Member Function Documentation

5.183.3.1 `_ThreadIndex __gnu_parallel::parallel_tag::__get_num_threads ()` `[inline]`

Find out desired number of threads.

Returns

Desired number of threads.

Definition at line 63 of file tags.h.

Referenced by `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::multiway_merge()`, and `__gnu_parallel::multiway_merge_sentinels()`.

5.183.3.2 `void __gnu_parallel::parallel_tag::set_num_threads (_ThreadIndex __num_threads)` `[inline]`

Set the desired number of threads.

Parameters

<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

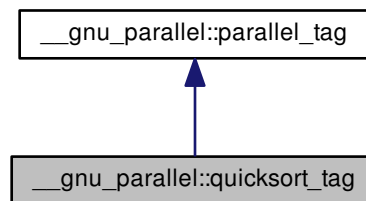
Definition at line 73 of file tags.h.

The documentation for this struct was generated from the following file:

- [tags.h](#)

5.184 `__gnu_parallel::quicksort_tag` Struct Reference

Inheritance diagram for `__gnu_parallel::quicksort_tag`:



Public Member Functions

- **`quicksort_tag`** (`_ThreadIndex` __num_threads)
- `_ThreadIndex` `__get_num_threads` ()
- void `set_num_threads` (`_ThreadIndex` __num_threads)

5.184.1 Detailed Description

Forces parallel sorting using unbalanced quicksort at compile time.

Definition at line 155 of file tags.h.

5.184.2 Member Function Documentation

5.184.2.1 `_ThreadIndex __gnu_parallel::parallel_tag::get_num_threads ()` `[inline]`, `[inherited]`

Find out desired number of threads.

Returns

Desired number of threads.

Definition at line 63 of file `tags.h`.

Referenced by `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::multiway_merge()`, and `__gnu_parallel::multiway_merge_sentinels()`.

5.184.2.2 `void __gnu_parallel::parallel_tag::set_num_threads (_ThreadIndex __num_threads)` `[inline]`, `[inherited]`

Set the desired number of threads.

Parameters

<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

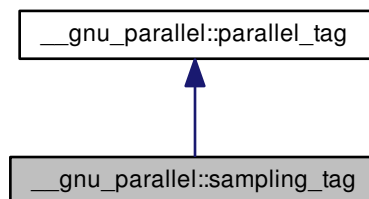
Definition at line 73 of file `tags.h`.

The documentation for this struct was generated from the following file:

- [tags.h](#)

5.185 `__gnu_parallel::sampling_tag` Struct Reference

Inheritance diagram for `__gnu_parallel::sampling_tag`:



Public Member Functions

- **sampling_tag** ([_ThreadIndex](#) __num_threads)
- [_ThreadIndex](#) **__get_num_threads** ()
- void **set_num_threads** ([_ThreadIndex](#) __num_threads)

5.185.1 Detailed Description

Forces parallel merging with exact splitting, at compile time.

Definition at line 118 of file tags.h.

5.185.2 Member Function Documentation

5.185.2.1 [_ThreadIndex](#) **__gnu_parallel::parallel_tag::__get_num_threads** () [\[inline\]](#), [\[inherited\]](#)

Find out desired number of threads.

Returns

Desired number of threads.

Definition at line 63 of file tags.h.

Referenced by [__gnu_parallel::__parallel_sort\(\)](#), [__gnu_parallel::multiway_merge\(\)](#), and [__gnu_parallel::multiway_merge_sentinels\(\)](#).

5.185.2.2 void **__gnu_parallel::parallel_tag::set_num_threads** ([_ThreadIndex](#) __num_threads) [\[inline\]](#), [\[inherited\]](#)

Set the desired number of threads.

Parameters

__num_threads	Desired number of threads.
-------------------------------	----------------------------

Definition at line 73 of file tags.h.

The documentation for this struct was generated from the following file:

- [tags.h](#)

5.186 [__gnu_parallel::sequential_tag](#) Struct Reference

5.186.1 Detailed Description

Forces sequential execution at compile time.

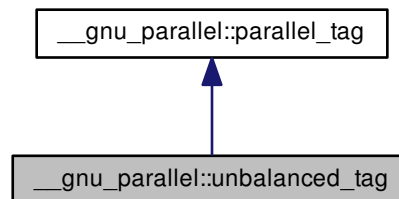
Definition at line 42 of file tags.h.

The documentation for this struct was generated from the following file:

- [tags.h](#)

5.187 `__gnu_parallel::unbalanced_tag` Struct Reference

Inheritance diagram for `__gnu_parallel::unbalanced_tag`:



Public Member Functions

- [_ThreadIndex __get_num_threads\(\)](#)
- [void set_num_threads\(_ThreadIndex __num_threads\)](#)

5.187.1 Detailed Description

Recommends parallel execution using static load-balancing at compile time.

Definition at line 92 of file tags.h.

5.187.2 Member Function Documentation

5.187.2.1 `_ThreadIndex __gnu_parallel::parallel_tag::__get_num_threads()` `[inline]`, `[inherited]`

Find out desired number of threads.

Returns

Desired number of threads.

Definition at line 63 of file tags.h.

Referenced by `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::multiway_merge()`, and `__gnu_parallel::multiway_merge_sentinels()`.

5.187.2.2 `void __gnu_parallel::parallel_tag::set_num_threads (_ThreadIndex __num_threads) [inline], [inherited]`

Set the desired number of threads.

Parameters

<code>__num_threads</code>	Desired number of threads.
----------------------------	----------------------------

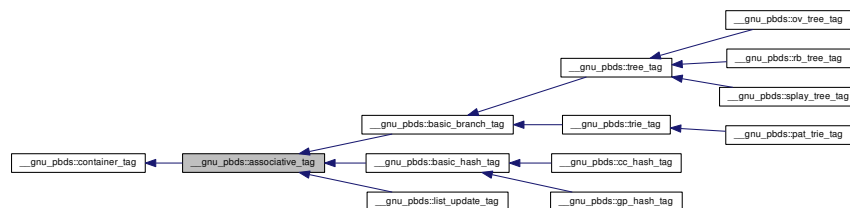
Definition at line 73 of file tags.h.

The documentation for this struct was generated from the following file:

- [tags.h](#)

5.188 __gnu_pbds::associative_tag Struct Reference

Inheritance diagram for __gnu_pbds::associative_tag:



5.188.1 Detailed Description

Basic associative-container.

Definition at line 135 of file tag_and_trait.hpp.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.189 __gnu_pbds::basic_branch< Key, Mapped, Tag, Node_Update, Policy_Tl, _Alloc > Class Template Reference

Inherits type< Key, Mapped, _Alloc, Tag, Policy_Tl >.

Public Types

- typedef `Node_Update` **node_update**

Protected Member Functions

- **basic_branch** (const [basic_branch](#) &other)
- template<typename T0 >
basic_branch (T0 t0)
- template<typename T0 , typename T1 >
basic_branch (T0 t0, T1 t1)
- template<typename T0 , typename T1 , typename T2 >
basic_branch (T0 t0, T1 t1, T2 t2)
- template<typename T0 , typename T1 , typename T2 , typename T3 >
basic_branch (T0 t0, T1 t1, T2 t2, T3 t3)
- template<typename T0 , typename T1 , typename T2 , typename T3 , typename T4 >
basic_branch (T0 t0, T1 t1, T2 t2, T3 t3, T4 t4)
- template<typename T0 , typename T1 , typename T2 , typename T3 , typename T4 , typename T5 >
basic_branch (T0 t0, T1 t1, T2 t2, T3 t3, T4 t4, T5 t5)
- template<typename T0 , typename T1 , typename T2 , typename T3 , typename T4 , typename T5 , typename T6 >
basic_branch (T0 t0, T1 t1, T2 t2, T3 t3, T4 t4, T5 t5, T6 t6)

5.189.1 Detailed Description

```
template<typename Key, typename Mapped, typename Tag, typename Node_Update, typename Policy_Tl, typename _Alloc>
class __gnu_pbds::basic_branch< Key, Mapped, Tag, Node_Update, Policy_Tl, _Alloc >
```

A branched, tree-like (tree, trie) container abstraction.

Template Parameters

<i>Key</i>	Key type.
<i>Mapped</i>	Map type.
<i>Tag</i>	Instantiating data structure type, see <code>container_tag</code> .
<i>Node_Update</i>	Updates nodes, restores invariants.
<i>Policy_Tl</i>	Policy typelist.
<i>_Alloc</i>	Allocator type.

Base is dispatched at compile time via `Tag`, from the following choices: `tree_tag`, `trie_tag`, and their descendants.

Base choices are: `detail::ov_tree_map`, `detail::rb_tree_map`, `detail::splay_tree_map`, and `detail::pat_trie_map`.

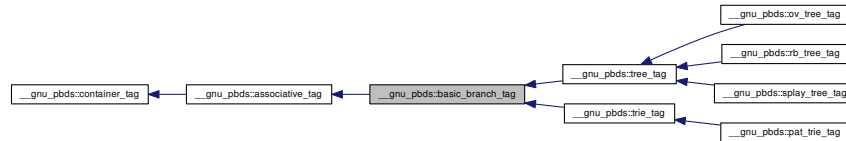
Definition at line 555 of file `assoc_container.hpp`.

The documentation for this class was generated from the following file:

- [assoc_container.hpp](#)

5.190 `__gnu_pbds::basic_branch_tag` Struct Reference

Inheritance diagram for `__gnu_pbds::basic_branch_tag`:



5.190.1 Detailed Description

Basic branch structure.

Definition at line 147 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.191 `__gnu_pbds::basic_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Resize_Policy, Store_Hash, Tag, Policy_Tl, _Alloc >` Class Template Reference

Inherits `type< Key, Mapped, _Alloc, Tag, __gnu_cxx::typelist::append< __gnu_cxx::typelist::create4< Hash_Fn, Eq_Fn, Resize_Policy, detail::integral_constant< int, Store_Hash > >::type, Policy_Tl >::type >`.

Protected Member Functions

- **`basic_hash_table`** (const [basic_hash_table](#) &other)
- `template<typename T0 >`
`basic_hash_table` (T0 t0)
- `template<typename T0 , typename T1 >`
`basic_hash_table` (T0 t0, T1 t1)
- `template<typename T0 , typename T1 , typename T2 >`
`basic_hash_table` (T0 t0, T1 t1, T2 t2)
- `template<typename T0 , typename T1 , typename T2 , typename T3 >`
`basic_hash_table` (T0 t0, T1 t1, T2 t2, T3 t3)
- `template<typename T0 , typename T1 , typename T2 , typename T3 , typename T4 >`
`basic_hash_table` (T0 t0, T1 t1, T2 t2, T3 t3, T4 t4)
- `template<typename T0 , typename T1 , typename T2 , typename T3 , typename T4 , typename T5 >`
`basic_hash_table` (T0 t0, T1 t1, T2 t2, T3 t3, T4 t4, T5 t5)
- `template<typename T0 , typename T1 , typename T2 , typename T3 , typename T4 , typename T5 , typename T6 >`
`basic_hash_table` (T0 t0, T1 t1, T2 t2, T3 t3, T4 t4, T5 t5, T6 t6)
- `template<typename T0 , typename T1 , typename T2 , typename T3 , typename T4 , typename T5 , typename T6 , typename T7 >`
`basic_hash_table` (T0 t0, T1 t1, T2 t2, T3 t3, T4 t4, T5 t5, T6 t6, T7 t7)
- `template<typename T0 , typename T1 , typename T2 , typename T3 , typename T4 , typename T5 , typename T6 , typename T7 , typename T8 >`
`basic_hash_table` (T0 t0, T1 t1, T2 t2, T3 t3, T4 t4, T5 t5, T6 t6, T7 t7, T8 t8)

5.191.1 Detailed Description

```
template<typename Key, typename Mapped, typename Hash_Fn, typename Eq_Fn, typename Resize_Policy, bool Store_Hash, type-
name Tag, typename Policy_Tl, typename _Alloc>
class __gnu_pbds::basic_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Resize_Policy, Store_Hash, Tag, Policy_Tl, _Alloc >
```

A hashed container abstraction.

Template Parameters

<i>Key</i>	Key type.
<i>Mapped</i>	Map type.
<i>Hash_Fn</i>	Hashing functor.
<i>Eq_Fn</i>	Equal functor.
<i>Resize_Policy</i>	Resizes hash.
<i>Store_Hash</i>	Indicates whether the hash value will be stored along with each key.
<i>Tag</i>	Instantiating data structure type, see <code>container_tag</code> .
<i>Policy_Tl</i>	Policy typelist.
<i>_Alloc</i>	Allocator type.

Base is dispatched at compile time via `Tag`, from the following choices: `cc_hash_tag`, `gp_hash_tag`, and descendants of `basic_hash_tag`.

Base choices are: `detail::cc_ht_map`, `detail::gp_ht_map`

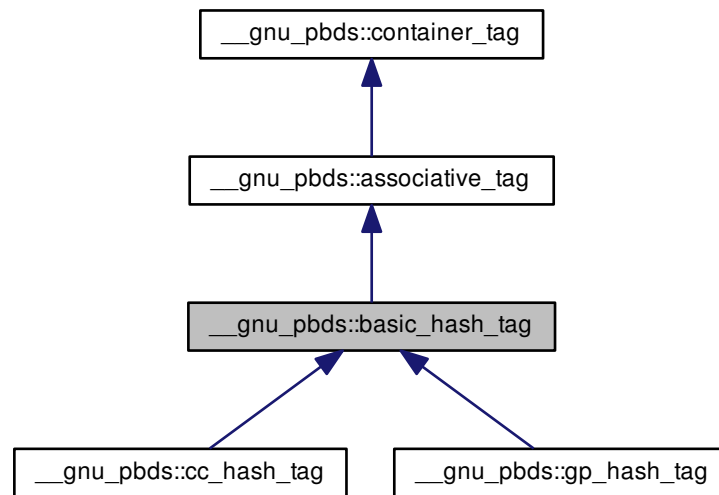
Definition at line 104 of file `assoc_container.hpp`.

The documentation for this class was generated from the following file:

- [assoc_container.hpp](#)

5.192 `__gnu_pbds::basic_hash_tag` Struct Reference

Inheritance diagram for `__gnu_pbds::basic_hash_tag`:



5.192.1 Detailed Description

Basic hash structure.

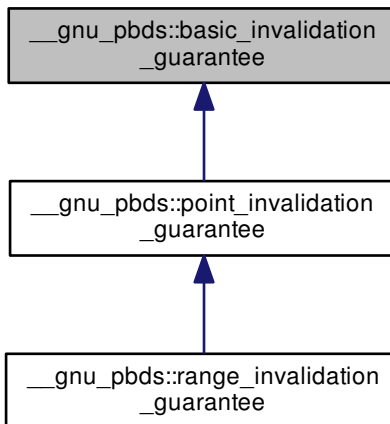
Definition at line 138 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.193 `__gnu_pbds::basic_invalidation_guarantee` Struct Reference

Inheritance diagram for `__gnu_pbds::basic_invalidation_guarantee`:



5.193.1 Detailed Description

Signifies a basic invalidation guarantee that any iterator, pointer, or reference to a container object's mapped value type is valid as long as the container is not modified.

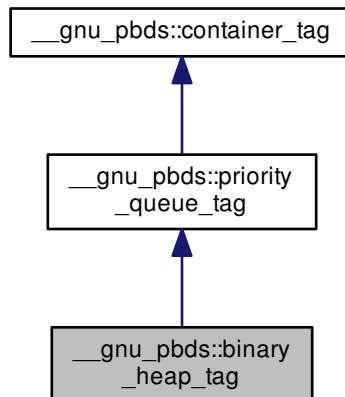
Definition at line 93 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.194 __gnu_pbds::binary_heap_tag Struct Reference

Inheritance diagram for __gnu_pbds::binary_heap_tag:



5.194.1 Detailed Description

Binary-heap (array-based).

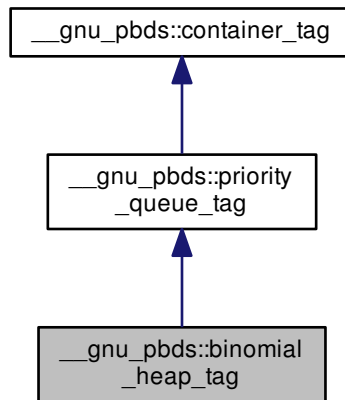
Definition at line 183 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.195 `__gnu_pbds::binomial_heap_tag` Struct Reference

Inheritance diagram for `__gnu_pbds::binomial_heap_tag`:



5.195.1 Detailed Description

Binomial-heap.

Definition at line 177 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.196 `__gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >`
Class Template Reference

Public Types

- enum { [external_load_access](#) }
- typedef `Size_Type` **size_type**

Public Member Functions

- [cc_hash_max_collision_check_resize_trigger](#) (float load=0.5)
- float [get_load](#) () const
- void [set_load](#) (float load)
- void **swap** ([cc_hash_max_collision_check_resize_trigger](#)< `External_Load_Access`, `Size_Type` > &other)

Protected Member Functions

- bool `is_grow_needed` (size_type size, size_type num_entries) const
- bool `is_resize_needed` () const
- void `notify_cleared` ()
- void `notify_erase_search_collision` ()
- void `notify_erase_search_end` ()
- void `notify_erase_search_start` ()
- void `notify_erased` (size_type num_entries)
- void `notify_externally_resized` (size_type new_size)
- void `notify_find_search_collision` ()
- void `notify_find_search_end` ()
- void `notify_find_search_start` ()
- void `notify_insert_search_collision` ()
- void `notify_insert_search_end` ()
- void `notify_insert_search_start` ()
- void `notify_inserted` (size_type num_entries)
- void `notify_resized` (size_type new_size)

5.196.1 Detailed Description

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
class __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >
```

A resize trigger policy based on collision checks. It keeps the simulated load factor lower than some given load factor.

Definition at line 293 of file hash_policy.hpp.

5.196.2 Member Enumeration Documentation

5.196.2.1 `template<bool External_Load_Access = false, typename Size_Type = std::size_t> anonymous enum`

Enumerator

`external_load_access` Specifies whether the load factor can be accessed externally. The two options have different trade-offs in terms of flexibility, genericity, and encapsulation.

Definition at line 298 of file hash_policy.hpp.

5.196.3 Constructor & Destructor Documentation

5.196.3.1 `template<bool External_Load_Access, typename Size_Type > __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >::cc_hash_max_collision_check_resize_trigger (float load = 0.5)`

Default constructor, or constructor taking load, a `__load` factor which it will attempt to maintain.

Definition at line 44 of file hash_policy.hpp.

5.196.4 Member Function Documentation

5.196.4.1 `template<bool External_Load_Access, typename Size_Type > float __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >::get_load () const`
[inline]

Returns the current load.

Definition at line 190 of file hash_policy.hpp.

5.196.4.2 `template<bool External_Load_Access, typename Size_Type > bool __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >::is_grow_needed (size_type size, size_type num_entries) const` [inline],[protected]

Queries whether a grow is needed. This method is called only if this object indicated is needed.

Definition at line 133 of file hash_policy.hpp.

5.196.4.3 `template<bool External_Load_Access, typename Size_Type > bool __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >::is_resize_needed () const` [inline],[protected]

Queries whether a resize is needed.

Definition at line 127 of file hash_policy.hpp.

5.196.4.4 `template<bool External_Load_Access, typename Size_Type > void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >::notify_cleared ()`
[protected]

Notifies the table was cleared.

Definition at line 121 of file hash_policy.hpp.

5.196.4.5 `template<bool External_Load_Access, typename Size_Type > void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >::notify_erase_search_collision ()` [inline],[protected]

Notifies a search encountered a collision.

Definition at line 97 of file hash_policy.hpp.

5.196.4.6 `template<bool External_Load_Access, typename Size_Type > void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >::notify_erase_search_end ()` [inline],[protected]

Notifies a search ended.

Definition at line 103 of file hash_policy.hpp.

5.196.4.7 `template<bool External_Load_Access, typename Size_Type > void __gnu_pbds::cc_hash_max_collision_↵
_check_resize_trigger< External_Load_Access, Size_Type >::notify_erase_search_start () [inline],
[protected]`

Notifies an erase search started.

Definition at line 91 of file hash_policy.hpp.

5.196.4.8 `template<bool External_Load_Access, typename Size_Type > void __gnu_pbds::cc_hash_max_collision_↵
_check_resize_trigger< External_Load_Access, Size_Type >::notify_erased (size_type num_entries)
[inline], [protected]`

Notifies an element was erased.

Definition at line 115 of file hash_policy.hpp.

5.196.4.9 `template<bool External_Load_Access, typename Size_Type > void __gnu_pbds::cc_hash_max_collision_↵
_check_resize_trigger< External_Load_Access, Size_Type >::notify_externally_resized (size_type new_size)
[protected]`

Notifies the table was resized externally.

Definition at line 172 of file hash_policy.hpp.

5.196.4.10 `template<bool External_Load_Access, typename Size_Type > void __gnu_pbds::cc_hash_max_collision_↵
_check_resize_trigger< External_Load_Access, Size_Type >::notify_find_search_collision () [inline],
[protected]`

Notifies a search encountered a collision.

Definition at line 61 of file hash_policy.hpp.

5.196.4.11 `template<bool External_Load_Access, typename Size_Type > void __gnu_pbds::cc_hash_max_collision_↵
_check_resize_trigger< External_Load_Access, Size_Type >::notify_find_search_end () [inline],
[protected]`

Notifies a search ended.

Definition at line 67 of file hash_policy.hpp.

5.196.4.12 `template<bool External_Load_Access, typename Size_Type > void __gnu_pbds::cc_hash_max_collision_↵
_check_resize_trigger< External_Load_Access, Size_Type >::notify_find_search_start () [inline],
[protected]`

Notifies a find search started.

Definition at line 55 of file hash_policy.hpp.

5.196.4.13 `template<bool External_Load_Access, typename Size_Type > void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >::notify_insert_search_collision () [inline], [protected]`

Notifies a search encountered a collision.

Definition at line 79 of file `hash_policy.hpp`.

5.196.4.14 `template<bool External_Load_Access, typename Size_Type > void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >::notify_insert_search_end () [inline], [protected]`

Notifies a search ended.

Definition at line 85 of file `hash_policy.hpp`.

5.196.4.15 `template<bool External_Load_Access, typename Size_Type > void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >::notify_insert_search_start () [inline], [protected]`

Notifies an insert search started.

Definition at line 73 of file `hash_policy.hpp`.

5.196.4.16 `template<bool External_Load_Access, typename Size_Type > void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >::notify_inserted (size_type num_entries) [inline], [protected]`

Notifies an element was inserted.

Definition at line 109 of file `hash_policy.hpp`.

5.196.4.17 `template<bool External_Load_Access, typename Size_Type > void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >::notify_resized (size_type new_size) [protected]`

Notifies the table was resized as a result of this object's signifying that a resize is needed.

Definition at line 139 of file `hash_policy.hpp`.

5.196.4.18 `template<bool External_Load_Access, typename Size_Type > void __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >::set_load (float load)`

Sets the load; does not resize the container.

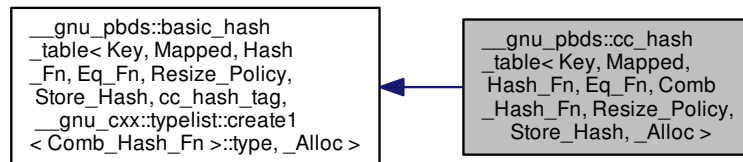
Definition at line 205 of file `hash_policy.hpp`.

The documentation for this class was generated from the following file:

- [hash_policy.hpp](#)

5.197 `__gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc >` Class Template Reference

Inheritance diagram for `__gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc >`:



Public Types

- typedef Comb_Hash_Fn **comb_hash_fn**
- typedef `cc_hash_tag` **container_category**
- typedef Eq_Fn **eq_fn**
- typedef Hash_Fn **hash_fn**
- typedef Resize_Policy **resize_policy**

Public Member Functions

- `cc_hash_table` ()
- `cc_hash_table` (const hash_fn &h)
- `cc_hash_table` (const hash_fn &h, const eq_fn &e)
- `cc_hash_table` (const hash_fn &h, const eq_fn &e, const comb_hash_fn &ch)
- `cc_hash_table` (const hash_fn &h, const eq_fn &e, const comb_hash_fn &ch, const resize_policy &rp)
- template<typename It >
`cc_hash_table` (It first, It last)
- template<typename It >
`cc_hash_table` (It first, It last, const hash_fn &h)
- template<typename It >
`cc_hash_table` (It first, It last, const hash_fn &h, const eq_fn &e)
- template<typename It >
`cc_hash_table` (It first, It last, const hash_fn &h, const eq_fn &e, const comb_hash_fn &ch)
- template<typename It >
`cc_hash_table` (It first, It last, const hash_fn &h, const eq_fn &e, const comb_hash_fn &ch, const resize_policy &rp)
- **cc_hash_table** (const `cc_hash_table` &other)
- `cc_hash_table` & **operator=** (const `cc_hash_table` &other)
- void **swap** (`cc_hash_table` &other)

5.197.1 Detailed Description

```
template<typename Key, typename Mapped, typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn
= typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn = detail::default_comb_hash_fn::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Hash_Fn>::type, bool Store_Hash = detail::default_store_hash, typename
_Alloc = std::allocator<char>>>
class __gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc >
```

A collision-chaining hash-based associative container.

Template Parameters

<i>Key</i>	Key type.
<i>Mapped</i>	Map type.
<i>Hash_Fn</i>	Hashing functor.
<i>Eq_Fn</i>	Equal functor.
<i>Comb_Hash_Fn</i>	Combining hash functor. If <i>Hash_Fn</i> is not null_type, then this is the ranged-hash functor; otherwise, this is the range-hashing functor. XXX(See Design::Hash-Based Containers::Hash Policies.)
<i>Resize_Policy</i>	Resizes hash.
<i>Store_Hash</i>	Indicates whether the hash value will be stored along with each key. If <i>Hash_Fn</i> is null_type, then the container will not compile if this value is true
<i>_Alloc</i>	Allocator type.

Base tag choices are: `cc_hash_tag`.

Base is `basic_hash_table`.

Definition at line 204 of file `assoc_container.hpp`.

5.197.2 Constructor & Destructor Documentation

5.197.2.1 `template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn = detail::default_comb_hash_fn::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Hash_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>> __gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc >::cc_hash_table () [inline]`

Default constructor.

Definition at line 217 of file `assoc_container.hpp`.

5.197.2.2 `template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn = detail::default_comb_hash_fn::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Hash_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>> __gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc >::cc_hash_table (const hash_fn & h) [inline]`

Constructor taking some policy objects. `r_hash_fn` will be copied by the `Hash_Fn` object of the container object.

Definition at line 221 of file `assoc_container.hpp`.

5.197.2.3 `template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn = detail::default_comb_hash_fn::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Hash_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>> __gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc >::cc_hash_table (const hash_fn & h, const eq_fn & e) [inline]`

Constructor taking some policy objects. `r_hash_fn` will be copied by the `hash_fn` object of the container object, and `r_eq_fn` will be copied by the `eq_fn` object of the container object.

Definition at line 228 of file `assoc_container.hpp`.

5.197.2.4 `template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn = detail::default_comb_hash_fn::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Hash_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>> __gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc >::cc_hash_table (const hash_fn & h, const eq_fn & e, const comb_hash_fn & ch) [inline]`

Constructor taking some policy objects. `r_hash_fn` will be copied by the `hash_fn` object of the container object, `r_eq_fn` will be copied by the `eq_fn` object of the container object, and `r_comb_hash_fn` will be copied by the `comb_hash_fn` object of the container object.

Definition at line 236 of file `assoc_container.hpp`.

5.197.2.5 `template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn = detail::default_comb_hash_fn::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Hash_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>> __gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc >::cc_hash_table (const hash_fn & h, const eq_fn & e, const comb_hash_fn & ch, const resize_policy & rp) [inline]`

Constructor taking some policy objects. `r_hash_fn` will be copied by the `hash_fn` object of the container object, `r_eq_fn` will be copied by the `eq_fn` object of the container object, `r_comb_hash_fn` will be copied by the `comb_hash_fn` object of the container object, and `r_resize_policy` will be copied by the `resize_policy` object of the container object.

Definition at line 245 of file `assoc_container.hpp`.

5.197.2.6 `template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn = detail::default_comb_hash_fn::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Hash_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>> template<typename It > __gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc >::cc_hash_table (It first, It last) [inline]`

Constructor taking `__iterators` to a range of `value_types`. The `value_types` between `first_it` and `last_it` will be inserted into the container object.

Definition at line 253 of file `assoc_container.hpp`.

```
5.197.2.7 template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type,
typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn = detail::default_comb_
_hash_fn::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Hash_Fn>::type, bool
Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>> template<typename It >
__gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc
>::cc_hash_table ( It first, It last, const hash_fn & h ) [inline]
```

Constructor taking `__iterators` to a range of `value_types` and some policy objects. The `value_types` between `first_it` and `last_it` will be inserted into the container object.

Definition at line 260 of file `assoc_container.hpp`.

```
5.197.2.8 template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type,
typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn = detail::default_comb_
_hash_fn::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Hash_Fn>::type, bool
Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>> template<typename It >
__gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc
>::cc_hash_table ( It first, It last, const hash_fn & h, const eq_fn & e ) [inline]
```

Constructor taking `__iterators` to a range of `value_types` and some policy objects The `value_types` between `first_it` and `last_it` will be inserted into the container object. `r_hash_fn` will be copied by the `hash_fn` object of the container object, and `r_eq_fn` will be copied by the `eq_fn` object of the container object.

Definition at line 271 of file `assoc_container.hpp`.

```
5.197.2.9 template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type,
typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn = detail::default_comb_
_hash_fn::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Hash_Fn>::type, bool
Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>> template<typename It >
__gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash, _Alloc
>::cc_hash_table ( It first, It last, const hash_fn & h, const eq_fn & e, const comb_hash_fn & ch ) [inline]
```

Constructor taking `__iterators` to a range of `value_types` and some policy objects The `value_types` between `first_it` and `last_it` will be inserted into the container object. `r_hash_fn` will be copied by the `hash_fn` object of the container object, `r_eq_fn` will be copied by the `eq_fn` object of the container object, and `r_comb_hash_fn` will be copied by the `comb_hash_fn` object of the container object.

Definition at line 283 of file `assoc_container.hpp`.

```
5.197.2.10 template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type,
typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Hash_Fn = detail::default_
comb_hash_fn::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Hash_Fn>::type,
bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>> template<typename It >
__gnu_pbds::cc_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_Hash,
_Alloc >::cc_hash_table ( It first, It last, const hash_fn & h, const eq_fn & e, const comb_hash_fn & ch, const
resize_policy & rp ) [inline]
```

Constructor taking `__iterators` to a range of `value_types` and some policy objects The `value_types` between `first_it` and `last_it` will be inserted into the container object. `r_hash_fn` will be copied by the `hash_fn` object of the container object, `r_eq_fn` will be copied by the `eq_fn` object of the container object, `r_comb_hash_fn` will be copied by the `comb_hash_fn` object of the container object, and `r_resize_policy` will be copied by the `resize_policy` object of the container object.

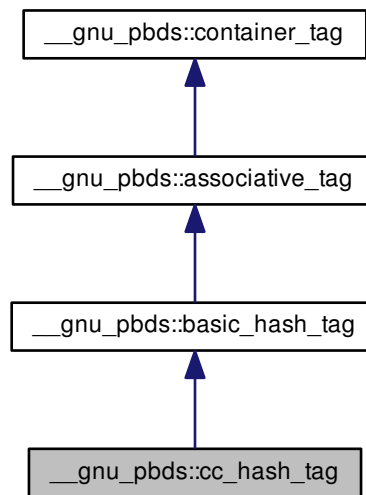
Definition at line 297 of file `assoc_container.hpp`.

The documentation for this class was generated from the following file:

- [assoc_container.hpp](#)

5.198 __gnu_pbds::cc_hash_tag Struct Reference

Inheritance diagram for __gnu_pbds::cc_hash_tag:



5.198.1 Detailed Description

Collision-chaining hash.

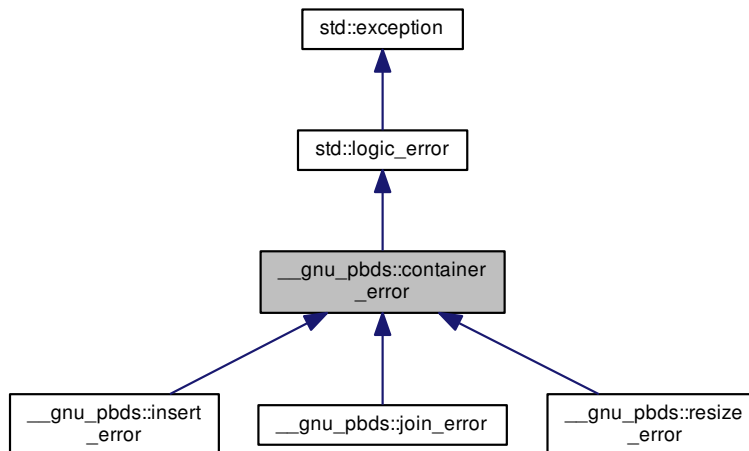
Definition at line 141 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.199 `__gnu_pbds::container_error` Struct Reference

Inheritance diagram for `__gnu_pbds::container_error`:



Public Member Functions

- virtual const char * [what](#) () const `_GLIBCXX_TXN_SAFE_DYN` noexcept

5.199.1 Detailed Description

Base class for exceptions.

Definition at line 57 of file `exception.hpp`.

5.199.2 Member Function Documentation

5.199.2.1 virtual const char* `std::logic_error::what` () const [virtual], [noexcept], [inherited]

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

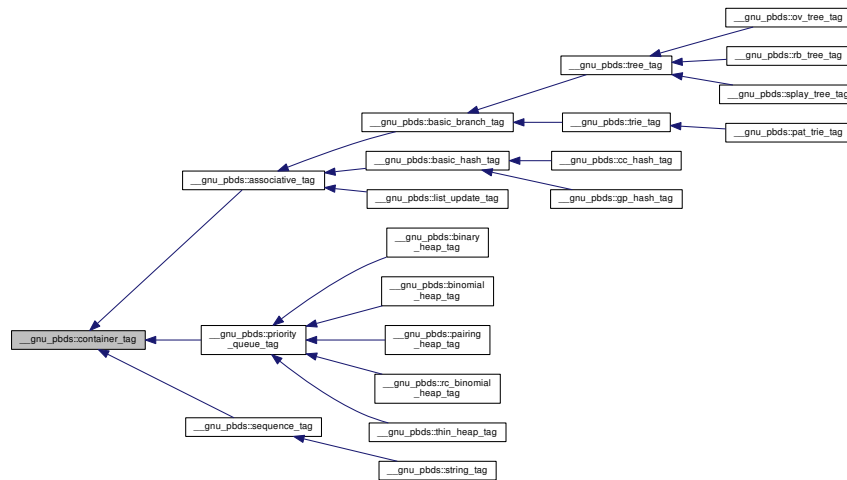
Reimplemented in [std::future_error](#).

The documentation for this struct was generated from the following file:

- [exception.hpp](#)

5.200 `__gnu_pbds::container_tag` Struct Reference

Inheritance diagram for `__gnu_pbds::container_tag`:



5.200.1 Detailed Description

Base data structure tag.

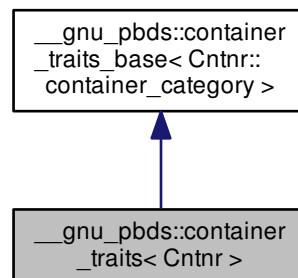
Definition at line 125 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.201 `__gnu_pbds::container_traits< Cntnr >` Struct Template Reference

Inheritance diagram for `__gnu_pbds::container_traits< Cntnr >`:



Public Types

- enum { `order_preserving`, `erase_can_throw`, `split_join_can_throw`, `reverse_iteration` }
- typedef `container_traits_base< container_category >` **base_type**
- typedef `Cntnr::container_category` **container_category**
- typedef `Cntnr` **container_type**
- typedef `base_type::invalidation_guarantee` **invalidation_guarantee**

5.201.1 Detailed Description

```
template<typename Cntnr>
struct __gnu_pbds::container_traits< Cntnr >
```

Container traits.

Definition at line 418 of file `tag_and_trait.hpp`.

5.201.2 Member Enumeration Documentation

5.201.2.1 `template<typename Cntnr>` anonymous enum

Enumerator

- order_preserving*** True only if `Cntnr` objects guarantee storing keys by order.
- erase_can_throw*** True only if erasing a key can throw.
- split_join_can_throw*** True only if split or join operations can throw.
- reverse_iteration*** True only reverse iterators are supported.

Definition at line 426 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.202 `__gnu_pbds::container_traits_base<_Tag>` Struct Template Reference

5.202.1 Detailed Description

```
template<typename _Tag>
struct __gnu_pbds::container_traits_base<_Tag>
```

Primary template, container traits base.

Definition at line 220 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.203 `__gnu_pbds::container_traits_base< binary_heap_tag >` Struct Template Reference

Public Types

- enum { **order_preserving**, **erase_can_throw**, **split_join_can_throw**, **reverse_iteration** }
- typedef [binary_heap_tag](#) **container_category**
- typedef [basic_invalidation_guarantee](#) **invalidation_guarantee**

5.203.1 Detailed Description

```
template<>
struct __gnu_pbds::container_traits_base< binary_heap_tag >
```

Specialization, binary heap.

Definition at line 400 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.204 `__gnu_pbds::container_traits_base< binomial_heap_tag >` Struct Template Reference

Public Types

- enum { **order_preserving**, **erase_can_throw**, **split_join_can_throw**, **reverse_iteration** }
- typedef [binomial_heap_tag](#) **container_category**
- typedef [point_invalidation_guarantee](#) **invalidation_guarantee**

5.204.1 Detailed Description

```
template<>
struct __gnu_pbds::container_traits_base< binomial_heap_tag >
```

Specialization, binomial heap.

Definition at line 368 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.205 `__gnu_pbds::container_traits_base< cc_hash_tag >` Struct Template Reference

Public Types

- enum { `order_preserving`, `erase_can_throw`, `split_join_can_throw`, `reverse_iteration` }
- typedef `cc_hash_tag` `container_category`
- typedef `point_invalidation_guarantee` `invalidation_guarantee`

5.205.1 Detailed Description

```
template<>
struct __gnu_pbds::container_traits_base< cc_hash_tag >
```

Specialization, cc hash.

Definition at line 224 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.206 `__gnu_pbds::container_traits_base< gp_hash_tag >` Struct Template Reference

Public Types

- enum { `order_preserving`, `erase_can_throw`, `split_join_can_throw`, `reverse_iteration` }
- typedef `gp_hash_tag` `container_category`
- typedef `basic_invalidation_guarantee` `invalidation_guarantee`

5.206.1 Detailed Description

```
template<>
struct __gnu_pbds::container_traits_base< gp_hash_tag >
```

Specialization, gp hash.

Definition at line 240 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.207 `__gnu_pbds::container_traits_base< list_update_tag >` Struct Template Reference

Public Types

- enum { **order_preserving**, **erase_can_throw**, **split_join_can_throw**, **reverse_iteration** }
- typedef [list_update_tag](#) **container_category**
- typedef [point_invalidation_guarantee](#) **invalidation_guarantee**

5.207.1 Detailed Description

```
template<>
struct __gnu_pbds::container_traits_base< list_update_tag >
```

Specialization, list update.

Definition at line 320 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.208 `__gnu_pbds::container_traits_base< ov_tree_tag >` Struct Template Reference

Public Types

- enum { **order_preserving**, **erase_can_throw**, **split_join_can_throw**, **reverse_iteration** }
- typedef [ov_tree_tag](#) **container_category**
- typedef [basic_invalidation_guarantee](#) **invalidation_guarantee**

5.208.1 Detailed Description

```
template<>
struct __gnu_pbds::container_traits_base< ov_tree_tag >
```

Specialization, ov tree.

Definition at line 288 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.209 `__gnu_pbds::container_traits_base< pairing_heap_tag >` Struct Template Reference

Public Types

- enum { `order_preserving`, `erase_can_throw`, `split_join_can_throw`, `reverse_iteration` }
- typedef `pairing_heap_tag` `container_category`
- typedef `point_invalidation_guarantee` `invalidation_guarantee`

5.209.1 Detailed Description

```
template<>
struct __gnu_pbds::container_traits_base< pairing_heap_tag >
```

Specialization, pairing heap.

Definition at line 336 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.210 `__gnu_pbds::container_traits_base< pat_trie_tag >` Struct Template Reference

Public Types

- enum { `order_preserving`, `erase_can_throw`, `split_join_can_throw`, `reverse_iteration` }
- typedef `pat_trie_tag` `container_category`
- typedef `range_invalidation_guarantee` `invalidation_guarantee`

5.210.1 Detailed Description

```
template<>
struct __gnu_pbds::container_traits_base< pat_trie_tag >
```

Specialization, pat trie.

Definition at line 304 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.211 `__gnu_pbds::container_traits_base< rb_tree_tag >` Struct Template Reference

Public Types

- enum { **order_preserving**, **erase_can_throw**, **split_join_can_throw**, **reverse_iteration** }
- typedef [rb_tree_tag](#) **container_category**
- typedef [range_invalidation_guarantee](#) **invalidation_guarantee**

5.211.1 Detailed Description

```
template<>
struct __gnu_pbds::container_traits_base< rb_tree_tag >
```

Specialization, rb tree.

Definition at line 256 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.212 `__gnu_pbds::container_traits_base< rc_binomial_heap_tag >` Struct Template Reference

Public Types

- enum { **order_preserving**, **erase_can_throw**, **split_join_can_throw**, **reverse_iteration** }
- typedef [rc_binomial_heap_tag](#) **container_category**
- typedef [point_invalidation_guarantee](#) **invalidation_guarantee**

5.212.1 Detailed Description

```
template<>
struct __gnu_pbds::container_traits_base< rc_binomial_heap_tag >
```

Specialization, rc binomial heap.

Definition at line 384 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.213 `__gnu_pbds::container_traits_base< splay_tree_tag >` Struct Template Reference

Public Types

- enum { `order_preserving`, `erase_can_throw`, `split_join_can_throw`, `reverse_iteration` }
- typedef `splay_tree_tag` `container_category`
- typedef `range_invalidation_guarantee` `invalidation_guarantee`

5.213.1 Detailed Description

```
template<>
struct __gnu_pbds::container_traits_base< splay_tree_tag >
```

Specialization, splay tree.

Definition at line 272 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.214 `__gnu_pbds::container_traits_base< thin_heap_tag >` Struct Template Reference

Public Types

- enum { `order_preserving`, `erase_can_throw`, `split_join_can_throw`, `reverse_iteration` }
- typedef `thin_heap_tag` `container_category`
- typedef `point_invalidation_guarantee` `invalidation_guarantee`

5.214.1 Detailed Description

```
template<>
struct __gnu_pbds::container_traits_base< thin_heap_tag >
```

Specialization, thin heap.

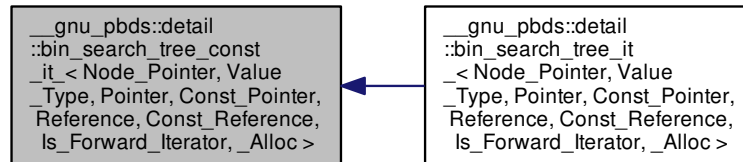
Definition at line 352 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.215 `__gnu_pbds::detail::bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >`:



Public Types

- typedef Const_Pointer **const_pointer**
- typedef Const_Reference **const_reference**
- typedef `_Alloc::difference_type` **difference_type**
- typedef `std::bidirectional_iterator_tag` **iterator_category**
- typedef Pointer **pointer**
- typedef Reference **reference**
- typedef Value_Type **value_type**

Public Member Functions

- **bin_search_tree_const_it_** (const Node_Pointer p_nd=0)
- **bin_search_tree_const_it_** (const [bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >](#) &other)
- bool **operator!=** (const [bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >](#) &other) const
- bool **operator!=** (const [bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >](#) &other) const
- const_reference **operator*** () const
- [bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >](#) & **operator++** ()
- [bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >](#) **operator++** (int)
- [bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >](#) & **operator--** ()
- [bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >](#) **operator--** (int)
- const_pointer **operator->** () const

- [bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >](#) & **operator=** (const [bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >](#) &other)
- [bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >](#) & **operator=** (const [bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >](#) &other)
- bool **operator==** (const [bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >](#) &other) const
- bool **operator==** (const [bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >](#) &other) const

Public Attributes

- Node_Pointer **m_p_nd**

Protected Member Functions

- void **dec** (false_type)
- void **dec** (true_type)
- void **inc** (false_type)
- void **inc** (true_type)

5.215.1 Detailed Description

template<typename Node_Pointer, typename Value_Type, typename Pointer, typename Const_Pointer, typename Reference, typename Const_Reference, bool Is_Forward_Iterator, typename _Alloc>
class `__gnu_pbds::detail::bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_`
`Reference, Is_Forward_Iterator, _Alloc >`

Const iterator.

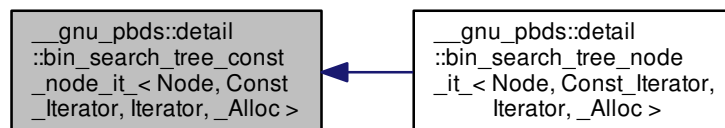
Definition at line 105 of file `point_iterators.hpp`.

The documentation for this class was generated from the following file:

- [point_iterators.hpp](#)

5.216 `__gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >`:



Public Types

- typedef Const_Iterator [const_reference](#)
- typedef [trivial_iterator_difference_type](#) difference_type
- typedef [trivial_iterator_tag](#) iterator_category
- typedef _Alloc::template rebind< [metadata_type](#) >::other::const_reference [metadata_const_reference](#)
- typedef Node::metadata_type [metadata_type](#)
- typedef Const_Iterator [reference](#)
- typedef Const_Iterator [value_type](#)

Public Member Functions

- **bin_search_tree_const_node_it_** (const node_pointer p_nd=0)
- [bin_search_tree_const_node_it_](#) < Node, Const_Iterator, Iterator, _Alloc > [get_l_child](#) () const
- [metadata_const_reference](#) [get_metadata](#) () const
- [bin_search_tree_const_node_it_](#) < Node, Const_Iterator, Iterator, _Alloc > [get_r_child](#) () const
- bool [operator!=](#) (const [bin_search_tree_const_node_it_](#) < Node, Const_Iterator, Iterator, _Alloc > &other) const
- [const_reference](#) [operator*](#) () const
- bool [operator==](#) (const [bin_search_tree_const_node_it_](#) < Node, Const_Iterator, Iterator, _Alloc > &other) const

Public Attributes

- node_pointer **m_p_nd**

5.216.1 Detailed Description

```
template<typename Node, class Const_Iterator, class Iterator, typename _Alloc>
class __gnu_pbds::detail::bin_search_tree_const_node_it_ < Node, Const_Iterator, Iterator, _Alloc >
```

Const node iterator.

Definition at line 58 of file bin_search_tree_/node_iterators.hpp.

5.216.2 Member Typedef Documentation

5.216.2.1 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > typedef Const_Iterator
__gnu_pbds::detail::bin_search_tree_const_node_it_ < Node, Const_Iterator, Iterator, _Alloc
>::const_reference`

Iterator's __const reference type.

Definition at line 80 of file bin_search_tree_/node_iterators.hpp.

5.216.2.2 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > typedef
trivial_iterator_difference_type __gnu_pbds::detail::bin_search_tree_const_node_it_< Node,
Const_Iterator, Iterator, _Alloc >::difference_type`

Difference type.

Definition at line 71 of file `bin_search_tree_/node_iterators.hpp`.

5.216.2.3 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > typedef trivial_iterator_tag
__gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc
>::iterator_category`

Category.

Definition at line 68 of file `bin_search_tree_/node_iterators.hpp`.

5.216.2.4 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > typedef _Alloc::template
rebind<metadata_type>::other::const_reference __gnu_pbds::detail::bin_search_tree_const_node_it_<
Node, Const_Iterator, Iterator, _Alloc >::metadata_const_reference`

Const metadata reference type.

Definition at line 88 of file `bin_search_tree_/node_iterators.hpp`.

5.216.2.5 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > typedef Node::metadata_type
__gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc
>::metadata_type`

Metadata type.

Definition at line 83 of file `bin_search_tree_/node_iterators.hpp`.

5.216.2.6 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > typedef Const_Iterator
__gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >::reference`

Iterator's reference type.

Definition at line 77 of file `bin_search_tree_/node_iterators.hpp`.

5.216.2.7 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > typedef Const_Iterator
__gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >::value_type`

Iterator's value type.

Definition at line 74 of file `bin_search_tree_/node_iterators.hpp`.

5.216.3 Member Function Documentation

5.216.3.1 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > bin_search_tree_const_node_↵
it<Node, Const_Iterator, Iterator, _Alloc> __gnu_pbds::detail::bin_search_tree_const_node_it_< Node,
Const_Iterator, Iterator, _Alloc >::get_l_child () const [inline]`

Returns the __const node iterator associated with the left node.

Definition at line 107 of file bin_search_tree_/node_iterators.hpp.

5.216.3.2 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > metadata_const_reference
__gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >::get_metadata
() const [inline]`

Metadata access.

Definition at line 102 of file bin_search_tree_/node_iterators.hpp.

5.216.3.3 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > bin_search_tree_const_node_↵
it<Node, Const_Iterator, Iterator, _Alloc> __gnu_pbds::detail::bin_search_tree_const_node_it_< Node,
Const_Iterator, Iterator, _Alloc >::get_r_child () const [inline]`

Returns the __const node iterator associated with the right node.

Definition at line 112 of file bin_search_tree_/node_iterators.hpp.

5.216.3.4 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > bool
__gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >::operator!= (
const bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc > & other) const [inline]`

Compares (negatively) to a different iterator object.

Definition at line 122 of file bin_search_tree_/node_iterators.hpp.

5.216.3.5 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > const_reference
__gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >::operator* ()
const [inline]`

Access.

Definition at line 97 of file bin_search_tree_/node_iterators.hpp.

5.216.3.6 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > bool
__gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >::operator== (
const bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc > & other) const [inline]`

Compares to a different iterator object.

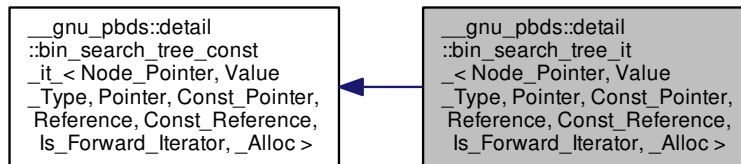
Definition at line 117 of file bin_search_tree_/node_iterators.hpp.

The documentation for this class was generated from the following file:

- [bin_search_tree_/node_iterators.hpp](#)

5.217 `__gnu_pbds::detail::bin_search_tree_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::bin_search_tree_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >`:



Public Types

- typedef Const_Pointer **const_pointer**
- typedef Const_Reference **const_reference**
- typedef `_Alloc::difference_type` **difference_type**
- typedef `std::bidirectional_iterator_tag` **iterator_category**
- typedef Pointer **pointer**
- typedef Reference **reference**
- typedef Value_Type **value_type**

Public Member Functions

- **bin_search_tree_it_** (const Node_Pointer p_nd=0)
- **bin_search_tree_it_** (const `bin_search_tree_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >` &other)
- bool **operator!=** (const `bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >` &other) const
- bool **operator!=** (const `bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >` &other) const
- `bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >::reference` **operator*** () const
- `bin_search_tree_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >` & **operator++** ()
- `bin_search_tree_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >` **operator++** (int)
- `bin_search_tree_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >` & **operator--** ()
- `bin_search_tree_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >` **operator--** (int)
- `bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >::pointer` **operator->** () const

- [bin_search_tree_it](#) < Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > & **operator=** (const [bin_search_tree_it](#) < Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > &other)
- [bin_search_tree_it](#) < Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > & **operator=** (const [bin_search_tree_it](#) < Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > &other)
- bool **operator==** (const [bin_search_tree_const_it](#) < Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > &other) const
- bool **operator==** (const [bin_search_tree_const_it](#) < Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > &other) const

Public Attributes

- Node_Pointer **m_p_nd**

Protected Types

- typedef [bin_search_tree_const_it](#) < Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc > **base_it_type**

Protected Member Functions

- void **dec** (false_type)
- void **dec** (true_type)
- void **inc** (false_type)
- void **inc** (true_type)

5.217.1 Detailed Description

```
template<typename Node_Pointer, typename Value_Type, typename Pointer, typename Const_Pointer, typename Reference, typename Const_Reference, bool Is_Forward_Iterator, typename _Alloc>
class __gnu_pbds::detail::bin_search_tree_it < Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >
```

Iterator.

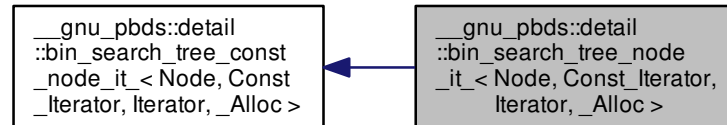
Definition at line 282 of file point_iterators.hpp.

The documentation for this class was generated from the following file:

- [point_iterators.hpp](#)

5.218 `__gnu_pbds::detail::bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc >` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc >`:



Public Types

- typedef Iterator [const_reference](#)
- typedef [trivial_iterator_difference_type](#) [difference_type](#)
- typedef [trivial_iterator_tag](#) [iterator_category](#)
- typedef `_Alloc::template rebind< metadata_type >::other::const_reference` [metadata_const_reference](#)
- typedef `Node::metadata_type` [metadata_type](#)
- typedef Iterator [reference](#)
- typedef Iterator [value_type](#)

Public Member Functions

- **`bin_search_tree_node_it_`** (const node_pointer p_nd=0)
- `bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc >` [get_l_child](#) () const
- `metadata_const_reference` [get_metadata](#) () const
- `bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc >` [get_r_child](#) () const
- bool [operator!=](#) (const `bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >` &other) const
- Iterator [operator*](#) () const
- bool [operator==](#) (const `bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >` &other) const

Public Attributes

- node_pointer **`m_p_nd`**

5.218.1 Detailed Description

```
template<typename Node, class Const_Iterator, class Iterator, typename _Alloc>
class __gnu_pbds::detail::bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc >
```

Node iterator.

Definition at line 136 of file `bin_search_tree_/node_iterators.hpp`.

5.218.2 Member Typedef Documentation

5.218.2.1 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > typedef Iterator
__gnu_pbds::detail::bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc >::const_reference`

Iterator's `__const` reference type.

Definition at line 153 of file `bin_search_tree_/node_iterators.hpp`.

5.218.2.2 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > typedef
trivial_iterator_difference_type __gnu_pbds::detail::bin_search_tree_const_node_it_< Node,
Const_Iterator, Iterator, _Alloc >::difference_type [inherited]`

Difference type.

Definition at line 71 of file `bin_search_tree_/node_iterators.hpp`.

5.218.2.3 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > typedef trivial_iterator_tag
__gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc
>::iterator_category [inherited]`

Category.

Definition at line 68 of file `bin_search_tree_/node_iterators.hpp`.

5.218.2.4 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > typedef _Alloc::template
rebind<metadata_type>::other::const_reference __gnu_pbds::detail::bin_search_tree_const_node_it_<
Node, Const_Iterator, Iterator, _Alloc >::metadata_const_reference [inherited]`

Const metadata reference type.

Definition at line 88 of file `bin_search_tree_/node_iterators.hpp`.

5.218.2.5 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > typedef Node::metadata_type
__gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc
>::metadata_type [inherited]`

Metadata type.

Definition at line 83 of file `bin_search_tree_/node_iterators.hpp`.

5.218.2.6 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > typedef Iterator
__gnu_pbds::detail::bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc >::reference`

Iterator's reference type.

Definition at line 150 of file `bin_search_tree_/node_iterators.hpp`.

5.218.2.7 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > typedef Iterator
__gnu_pbds::detail::bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc >::value_type`

Iterator's value type.

Definition at line 147 of file `bin_search_tree_/node_iterators.hpp`.

5.218.3 Member Function Documentation

5.218.3.1 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > bin_search_tree_node_it_↵
<Node, Const_Iterator, Iterator, _Alloc> __gnu_pbds::detail::bin_search_tree_node_it_< Node, Const_Iterator,
Iterator, _Alloc >::get_l_child () const [inline]`

Returns the node iterator associated with the left node.

Definition at line 167 of file `bin_search_tree_/node_iterators.hpp`.

5.218.3.2 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > metadata_const_reference
__gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >::get_metadata
() const [inline],[inherited]`

Metadata access.

Definition at line 102 of file `bin_search_tree_/node_iterators.hpp`.

5.218.3.3 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > bin_search_tree_node_it_↵
<Node, Const_Iterator, Iterator, _Alloc> __gnu_pbds::detail::bin_search_tree_node_it_< Node, Const_Iterator,
Iterator, _Alloc >::get_r_child () const [inline]`

Returns the node iterator associated with the right node.

Definition at line 175 of file `bin_search_tree_/node_iterators.hpp`.

5.218.3.4 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > bool
__gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >::operator!= (
const bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc > & other) const [inline],
[inherited]`

Compares (negatively) to a different iterator object.

Definition at line 122 of file `bin_search_tree_/node_iterators.hpp`.

5.218.3.5 `template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > Iterator
__gnu_pbds::detail::bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc >::operator*() const
[inline]`

Access.

Definition at line 162 of file `bin_search_tree_/node_iterators.hpp`.


```
5.218.3.6 template<typename Node , class Const_Iterator , class Iterator , typename _Alloc > bool
    __gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >::operator==(
        const bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc > & other ) const    [inline],
        [inherited]
```

Compares to a different iterator object.

Definition at line 117 of file bin_search_tree_/node_iterators.hpp.

The documentation for this class was generated from the following file:

- [bin_search_tree_/node_iterators.hpp](#)

5.219 __gnu_pbds::detail::bin_search_tree_traits< Key, Mapped, Cmp_Fn, Node_Update, Node, _Alloc > Struct Template Reference

Public Types

- typedef [bin_search_tree_const_it_](#)< typename _Alloc::template rebind< node >::other::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, false, _Alloc > **const_reverse_iterator**
- typedef Node **node**
- typedef [bin_search_tree_const_node_it_](#)< Node, [point_const_iterator](#), [point_iterator](#), _Alloc > [node_const_iterator](#)
- typedef [bin_search_tree_node_it_](#)< Node, [point_const_iterator](#), [point_iterator](#), _Alloc > **node_iterator**
- typedef Node_Update< [node_const_iterator](#), [node_iterator](#), Cmp_Fn, _Alloc > **node_update**
- typedef [__gnu_pbds::null_node_update](#)< [node_const_iterator](#), [node_iterator](#), Cmp_Fn, _Alloc > * **null_node_update_pointer**
- typedef [bin_search_tree_const_it_](#)< typename _Alloc::template rebind< node >::other::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, true, _Alloc > **point_const_iterator**
- typedef [bin_search_tree_it_](#)< typename _Alloc::template rebind< node >::other::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, true, _Alloc > **point_iterator**
- typedef [bin_search_tree_it_](#)< typename _Alloc::template rebind< node >::other::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, false, _Alloc > **reverse_iterator**

5.219.1 Detailed Description

```
template<typename Key, typename Mapped, class Cmp_Fn, template< typename Node_Cltr, class Node_Itr, class _Cmp_Fn, typename
_Alloc > class Node_Update, class Node, typename _Alloc>
struct __gnu_pbds::detail::bin_search_tree_traits< Key, Mapped, Cmp_Fn, Node_Update, Node, _Alloc >
```

Binary search tree traits, primary template.

Definition at line 63 of file bin_search_tree_/traits.hpp.

5.219.2 Member Typedef Documentation

5.219.2.1 `template<typename Key, typename Mapped, class Cmp_Fn, template< typename Node_Cltr, class Node_Itr, class _Cmp_Fn, typename _Alloc > class Node_Update, class Node, typename _Alloc> typedef bin_search_tree_const_node_it_< Node, point_const_iterator, point_iterator, _Alloc> __gnu_pbds::detail::bin_search_tree_traits< Key, Mapped, Cmp_Fn, Node_Update, Node, _Alloc >::node_const_iterator`

This is an iterator to an iterator: it iterates over nodes, and de-referencing it returns one of the tree's iterators.

Definition at line 131 of file `bin_search_tree_/traits.hpp`.

The documentation for this struct was generated from the following file:

- [bin_search_tree_/traits.hpp](#)

5.220 `__gnu_pbds::detail::bin_search_tree_traits`< Key, null_type, Cmp_Fn, Node_Update, Node, _Alloc > Struct Template Reference

Public Types

- typedef `bin_search_tree_const_it_< typename _Alloc::template rebind< node >::other::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, false, _Alloc >` **const_reverse_iterator**
- typedef Node **node**
- typedef `bin_search_tree_const_node_it_< Node, point_const_iterator, point_iterator, _Alloc >` **node_const_iterator**
- typedef `node_const_iterator` **node_iterator**
- typedef Node_Update< `node_const_iterator`, `node_iterator`, Cmp_Fn, _Alloc > **node_update**
- typedef `__gnu_pbds::null_node_update< node_const_iterator, node_iterator, Cmp_Fn, _Alloc > * null_node_update_pointer`
- typedef `bin_search_tree_const_it_< typename _Alloc::template rebind< node >::other::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, true, _Alloc >` **point_const_iterator**
- typedef `point_const_iterator` **point_iterator**
- typedef `const_reverse_iterator` **reverse_iterator**

5.220.1 Detailed Description

```
template<typename Key, class Cmp_Fn, template< typename Node_Cltr, class Node_Itr, class _Cmp_Fn, typename _Alloc > class
Node_Update, class Node, typename _Alloc>
struct __gnu_pbds::detail::bin_search_tree_traits< Key, null_type, Cmp_Fn, Node_Update, Node, _Alloc >
```

Specialization.

Definition at line 169 of file `bin_search_tree_/traits.hpp`.

5.220.2 Member Typedef Documentation

5.220.2.1 `template<typename Key , class Cmp_Fn , template< typename Node_Cltr, class Node_Itr, class _Cmp_Fn, typename _Alloc > class Node_Update, class Node , typename _Alloc > typedef bin_search_tree_const_node_it_< Node, point_const_iterator, point_iterator, _Alloc> __gnu_pbds::detail::bin_search_tree_traits< Key, null_type, Cmp_Fn, Node_Update, Node, _Alloc >::node_const_iterator`

This is an iterator to an iterator: it iterates over nodes, and de-referencing it returns one of the tree's iterators.

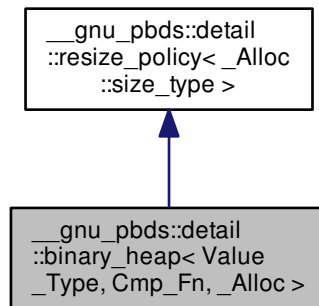
Definition at line 216 of file `bin_search_tree_/traits.hpp`.

The documentation for this struct was generated from the following file:

- [bin_search_tree_/traits.hpp](#)

5.221 __gnu_pbds::detail::binary_heap< Value_Type, Cmp_Fn, _Alloc > Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::binary_heap< Value_Type, Cmp_Fn, _Alloc >`:



Public Types

- typedef `_Alloc` **allocator_type**
- typedef `Cmp_Fn` **cmp_fn**
- typedef `cond_dealtor< value_type, _Alloc >` **cond_dealtor_t**
- typedef `binary_heap_const_iterator< value_type, entry, simple_value, _Alloc >` **const_iterator**
- typedef `value_allocator::const_pointer` **const_pointer**
- typedef `value_allocator::const_reference` **const_reference**
- typedef `_Alloc::difference_type` **difference_type**
- typedef `__conditional_type< simple_value, value_type, pointer >::__type` **entry**
- typedef `_Alloc::template rebind< entry >::other` **entry_allocator**
- typedef `entry_cmp< Value_Type, Cmp_Fn, _Alloc, is_simple< Value_Type >::value >::type` **entry_cmp**

- typedef `entry_allocator::pointer` **entry_pointer**
- typedef `const_iterator` **iterator**
- typedef `binary_heap_point_const_iterator< value_type, entry, simple_value, _Alloc >` **point_const_iterator**
- typedef `point_const_iterator` **point_iterator**
- typedef `value_allocator::pointer` **pointer**
- typedef `value_allocator::reference` **reference**
- typedef `__gnu_pbds::detail::resize_policy< typename _Alloc::size_type >` **resize_policy**
- typedef `_Alloc::size_type` **size_type**
- typedef `Value_Type` **value_type**

Public Member Functions

- **binary_heap** (const `cmp_fn` &)
- **binary_heap** (const `binary_heap` &)
- `iterator` **begin** ()
- `const_iterator` **begin** () const
- void **clear** ()
- bool **empty** () const
- `iterator` **end** ()
- `const_iterator` **end** () const
- void **erase** (`point_iterator`)
- void **erase_at** (`entry_pointer`, `size_type`, `false_type`)
- void **erase_at** (`entry_pointer`, `size_type`, `true_type`)
- template<typename `Pred` >
 `size_type` **erase_if** (`Pred`)
- `Cmp_Fn` & **get_cmp_fn** ()
- const `Cmp_Fn` & **get_cmp_fn** () const
- `size_type` **get_new_size_for_arbitrary** (`size_type`) const
- `size_type` **get_new_size_for_grow** () const
- `size_type` **get_new_size_for_shrink** () const
- bool **grow_needed** (`size_type`) const
- void **join** (`binary_heap` &)
- `size_type` **max_size** () const
- void **modify** (`point_iterator`, const `reference`)
- void **notify_arbitrary** (`size_type`)
- void **notify_grow_resize** ()
- void **notify_shrink_resize** ()
- void **pop** ()
- `point_iterator` **push** (const `reference`)
- bool **resize_needed_for_grow** (`size_type`) const
- bool **resize_needed_for_shrink** (`size_type`) const
- bool **shrink_needed** (`size_type`) const
- `size_type` **size** () const
- template<typename `Pred` >
 void **split** (`Pred`, `binary_heap` &)
- void **swap** (`resize_policy< _Alloc::size_type >` &)
- void **swap** (`binary_heap` &)
- const `reference` **top** () const

Static Public Attributes

- static const `_Alloc::size_type` **min_size**

Protected Member Functions

- template<typename It >
void **copy_from_range** (It, It)

5.221.1 Detailed Description

```
template<typename Value_Type, typename Cmp_Fn, typename _Alloc>
class __gnu_pbds::detail::binary_heap< Value_Type, Cmp_Fn, _Alloc >
```

Binary heaps composed of resize and compare policies.

Based on CLRS.

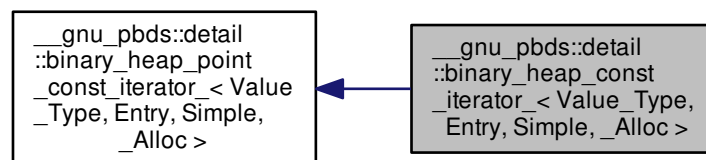
Definition at line 84 of file `binary_heap.hpp`.

The documentation for this class was generated from the following file:

- [binary_heap.hpp](#)

5.222 `__gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc >` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc >`:



Public Types

- typedef `base_type::const_pointer` `const_pointer`
- typedef `base_type::const_reference` `const_reference`
- typedef `_Alloc::difference_type` `difference_type`
- typedef `std::forward_iterator_tag` `iterator_category`
- typedef `base_type::pointer` `pointer`
- typedef `base_type::reference` `reference`
- typedef `base_type::value_type` `value_type`

Public Member Functions

- `binary_heap_const_iterator_` (`entry_pointer p_e`)
- `binary_heap_const_iterator_` ()
- `binary_heap_const_iterator_` (`const binary_heap_const_iterator_ &other`)
- `bool operator!=` (`const binary_heap_const_iterator_ &other`) `const`
- `bool operator!=` (`const binary_heap_point_const_iterator_ &other`) `const`
- `const_reference operator*` () `const`
- `binary_heap_const_iterator_ & operator++` ()
- `binary_heap_const_iterator_ operator++` (`int`)
- `const_pointer operator->` () `const`
- `bool operator==` (`const binary_heap_const_iterator_ &other`) `const`
- `bool operator==` (`const binary_heap_point_const_iterator_ &other`) `const`

Public Attributes

- `entry_pointer m_p_e`

5.222.1 Detailed Description

```
template<typename Value_Type, typename Entry, bool Simple, typename _Alloc>
class __gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc >
```

Const point-type iterator.

Definition at line 60 of file `binary_heap_/const_iterator.hpp`.

5.222.2 Member Typedef Documentation

5.222.2.1 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > typedef base_type::const_pointer __gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc >::const_pointer`

Iterator's const pointer type.

Definition at line 80 of file `binary_heap_/const_iterator.hpp`.

5.222.2.2 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > typedef
base_type::const_reference __gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry,
Simple, _Alloc >::const_reference`

Iterator's const reference type.

Definition at line 86 of file `binary_heap_/const_iterator.hpp`.

5.222.2.3 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > typedef _Alloc::difference_type
__gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc >::difference_type`

Difference type.

Definition at line 71 of file `binary_heap_/const_iterator.hpp`.

5.222.2.4 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > typedef std::forward_iterator_tag
__gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc >::iterator_category`

Category.

Definition at line 68 of file `binary_heap_/const_iterator.hpp`.

5.222.2.5 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > typedef base_type::pointer
__gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc >::pointer`

Iterator's pointer type.

Definition at line 77 of file `binary_heap_/const_iterator.hpp`.

5.222.2.6 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > typedef base_type::reference
__gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc >::reference`

Iterator's reference type.

Definition at line 83 of file `binary_heap_/const_iterator.hpp`.

5.222.2.7 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > typedef base_type::value_type
__gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc >::value_type`

Iterator's value type.

Definition at line 74 of file `binary_heap_/const_iterator.hpp`.

5.222.3 Constructor & Destructor Documentation

5.222.3.1 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > __gnu_pbds::detail::
::binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc >::binary_heap_const_iterator_ ()
[inline]`

Default constructor.

Definition at line 94 of file `binary_heap_/const_iterator.hpp`.

5.222.3.2 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > __gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc >::binary_heap_const_iterator_ (const binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc > & other) [inline]`

Copy constructor.

Definition at line 99 of file `binary_heap_/const_iterator.hpp`.

5.222.4 Member Function Documentation

5.222.4.1 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > bool __gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc >::operator!= (const binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc > & other) const [inline]`

Compares content (negatively) to a different iterator object.

Definition at line 110 of file `binary_heap_/const_iterator.hpp`.

5.222.4.2 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > bool __gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >::operator!= (const binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc > & other) const [inline], [inherited]`

Compares content (negatively) to a different iterator object.

Definition at line 126 of file `binary_heap_/point_const_iterator.hpp`.

5.222.4.3 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > const_reference __gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >::operator* () const [inline], [inherited]`

Access.

Definition at line 113 of file `binary_heap_/point_const_iterator.hpp`.

5.222.4.4 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > const_pointer __gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >::operator-> () const [inline], [inherited]`

Access.

Definition at line 105 of file `binary_heap_/point_const_iterator.hpp`.

5.222.4.5 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > bool __gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc >::operator== (const binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc > & other) const [inline]`

Compares content to a different iterator object.

Definition at line 105 of file `binary_heap_/const_iterator.hpp`.

5.222.4.6 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > bool
 __gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >::operator== (
 const binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc > & other) const [inline],
 [inherited]`

Compares content to a different iterator object.

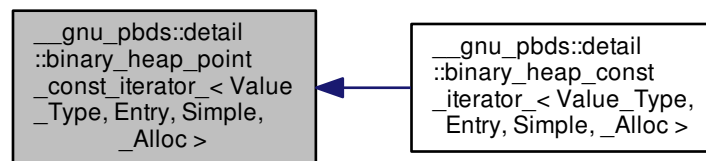
Definition at line 121 of file `binary_heap_/point_const_iterator.hpp`.

The documentation for this class was generated from the following file:

- [binary_heap_/const_iterator.hpp](#)

5.223 `__gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >`:



Public Types

- `typedef _Alloc::template rebind< value_type >::other::const_pointer const_pointer`
- `typedef _Alloc::template rebind< value_type >::other::const_reference const_reference`
- `typedef trivial_iterator_difference_type difference_type`
- `typedef trivial_iterator_tag iterator_category`
- `typedef _Alloc::template rebind< value_type >::other::pointer pointer`
- `typedef _Alloc::template rebind< value_type >::other::reference reference`
- `typedef Value_Type value_type`

Public Member Functions

- `binary_heap_point_const_iterator_ (entry_pointer p_e)`
- `binary_heap_point_const_iterator_ ()`
- `binary_heap_point_const_iterator_ (const binary_heap_point_const_iterator_ &other)`
- `bool operator!= (const binary_heap_point_const_iterator_ &other) const`
- `const_reference operator* () const`
- `const_pointer operator-> () const`
- `bool operator== (const binary_heap_point_const_iterator_ &other) const`

Public Attributes

- entry_pointer `m_p_e`

Protected Types

- `typedef _Alloc::template rebind< Entry >::other::pointer` `entry_pointer`

5.223.1 Detailed Description

```
template<typename Value_Type, typename Entry, bool Simple, typename _Alloc>
class __gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >
```

Const point-type iterator.

Definition at line 55 of file `binary_heap_/point_const_iterator.hpp`.

5.223.2 Member Typedef Documentation

5.223.2.1 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > typedef _Alloc::template rebind<value_type>::other::const_pointer` `__gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >::const_pointer`

Iterator's const pointer type.

Definition at line 77 of file `binary_heap_/point_const_iterator.hpp`.

5.223.2.2 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > typedef _Alloc::template rebind<value_type>::other::const_reference` `__gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >::const_reference`

Iterator's const reference type.

Definition at line 87 of file `binary_heap_/point_const_iterator.hpp`.

5.223.2.3 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > typedef trivial_iterator_difference_type` `__gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >::difference_type`

Difference type.

Definition at line 65 of file `binary_heap_/point_const_iterator.hpp`.

5.223.2.4 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > typedef trivial_iterator_tag
__gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc
>::iterator_category`

Category.

Definition at line 62 of file `binary_heap_/point_const_iterator.hpp`.

5.223.2.5 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > typedef _Alloc::template
rebind<value_type>::other::pointer __gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type,
Entry, Simple, _Alloc >::pointer`

Iterator's pointer type.

Definition at line 72 of file `binary_heap_/point_const_iterator.hpp`.

5.223.2.6 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > typedef _Alloc::template
rebind<value_type>::other::reference __gnu_pbds::detail::binary_heap_point_const_iterator_<
Value_Type, Entry, Simple, _Alloc >::reference`

Iterator's reference type.

Definition at line 82 of file `binary_heap_/point_const_iterator.hpp`.

5.223.2.7 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > typedef Value_Type
__gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >::value_type`

Iterator's value type.

Definition at line 68 of file `binary_heap_/point_const_iterator.hpp`.

5.223.3 Constructor & Destructor Documentation

5.223.3.1 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > __gnu_pbds::detail::binary_↵
heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >::binary_heap_point_const_iterator_()
[inline]`

Default constructor.

Definition at line 95 of file `binary_heap_/point_const_iterator.hpp`.

5.223.3.2 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > __gnu_pbds::detail::binary_↵
heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >::binary_heap_point_const_iterator_(
const binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc > & other) [inline]`

Copy constructor.

Definition at line 99 of file `binary_heap_/point_const_iterator.hpp`.

5.223.4 Member Function Documentation

5.223.4.1 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > bool
__gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >::operator!= (`
`const binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc > & other) const [inline]`

Compares content (negatively) to a different iterator object.

Definition at line 126 of file `binary_heap_/point_const_iterator.hpp`.

5.223.4.2 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > const_reference
__gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >::operator* ()`
`const [inline]`

Access.

Definition at line 113 of file `binary_heap_/point_const_iterator.hpp`.

5.223.4.3 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > const_pointer
__gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >::operator-> (`
`) const [inline]`

Access.

Definition at line 105 of file `binary_heap_/point_const_iterator.hpp`.

5.223.4.4 `template<typename Value_Type , typename Entry , bool Simple, typename _Alloc > bool
__gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >::operator== (`
`const binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc > & other) const [inline]`

Compares content to a different iterator object.

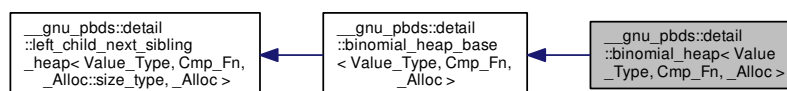
Definition at line 121 of file `binary_heap_/point_const_iterator.hpp`.

The documentation for this class was generated from the following file:

- [binary_heap_/point_const_iterator.hpp](#)

5.224 `__gnu_pbds::detail::binomial_heap< Value_Type, Cmp_Fn, _Alloc >` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::binomial_heap< Value_Type, Cmp_Fn, _Alloc >`:



Public Types

- typedef base_type::allocator_type **allocator_type**
- typedef base_type::cmp_fn **cmp_fn**
- typedef base_type::const_iterator **const_iterator**
- typedef base_type::const_pointer **const_pointer**
- typedef base_type::const_reference **const_reference**
- typedef _Alloc::difference_type **difference_type**
- typedef base_type::iterator **iterator**
- typedef base_type::point_const_iterator **point_const_iterator**
- typedef base_type::point_iterator **point_iterator**
- typedef base_type::pointer **pointer**
- typedef base_type::reference **reference**
- typedef _Alloc::size_type **size_type**
- typedef Value_Type **value_type**

Public Member Functions

- **binomial_heap** (const Cmp_Fn &)
- **binomial_heap** (const binomial_heap &)
- **iterator begin** ()
- **const_iterator begin** () const
- void **clear** ()
- bool **empty** () const
- **iterator end** ()
- **const_iterator end** () const
- void **erase** (point_iterator)
- template<typename Pred >
size_type **erase_if** (Pred)
- Cmp_Fn & **get_cmp_fn** ()
- const Cmp_Fn & **get_cmp_fn** () const
- void **join** (binomial_heap_base< Value_Type, Cmp_Fn, _Alloc > &)
- size_type **max_size** () const
- void **modify** (point_iterator, const_reference)
- void **pop** ()
- **point_iterator push** (const_reference)
- size_type **size** () const
- template<typename Pred >
void **split** (Pred, binomial_heap_base< Value_Type, Cmp_Fn, _Alloc > &)
- void **swap** (left_child_next_sibling_heap< Value_Type, Cmp_Fn, _Alloc::size_type, _Alloc > &)
- const_reference **top** () const

Protected Types

- typedef base_type::node **node**
- typedef _Alloc::template rebind< left_child_next_sibling_heap_node_ < Value_Type, _Alloc::size_type, _Alloc > ::other **node_allocator**
- typedef _Alloc::size_type **node_metadata**
- typedef std::pair< node_pointer, node_pointer > **node_pointer_pair**

Protected Member Functions

- void **actual_erase_node** (node_pointer)
- void **bubble_to_top** (node_pointer)
- void **clear_imp** (node_pointer)
- template<typename It >
void **copy_from_range** (It, It)
- void **find_max** ()
- node_pointer **get_new_node_for_insert** (const_reference)
- node_pointer **prune** (Pred)
- void **swap** ([binomial_heap_base](#)< Value_Type, Cmp_Fn, _Alloc > &)
- void **swap_with_parent** (node_pointer, node_pointer)
- void **to_linked_list** ()
- void **value_swap** ([left_child_next_sibling_heap](#) &)

Static Protected Member Functions

- static void **make_child_of** (node_pointer, node_pointer)
- static node_pointer **parent** (node_pointer)

Protected Attributes

- node_pointer **m_p_max**
- node_pointer **m_p_root**
- size_type **m_size**

5.224.1 Detailed Description

```
template<typename Value_Type, typename Cmp_Fn, typename _Alloc>
class __gnu_pbds::detail::binomial_heap< Value_Type, Cmp_Fn, _Alloc >
```

Binomial heap.

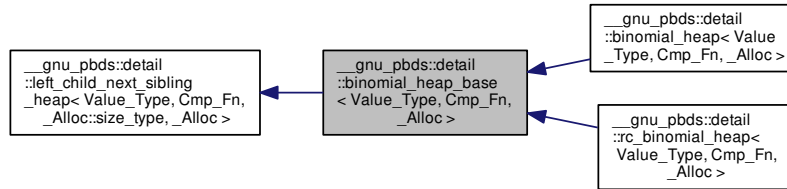
Definition at line 68 of file `binomial_heap_.hpp`.

The documentation for this class was generated from the following file:

- [binomial_heap_.hpp](#)

5.225 `__gnu_pbds::detail::binomial_heap_base< Value_Type, Cmp_Fn, _Alloc >` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::binomial_heap_base< Value_Type, Cmp_Fn, _Alloc >`:



Public Types

- typedef `_Alloc` **allocator_type**
- typedef `Cmp_Fn` **cmp_fn**
- typedef `base_type::const_iterator` **const_iterator**
- typedef `__rebind_v::const_pointer` **const_pointer**
- typedef `__rebind_v::const_reference` **const_reference**
- typedef `_Alloc::difference_type` **difference_type**
- typedef `base_type::iterator` **iterator**
- typedef `base_type::point_const_iterator` **point_const_iterator**
- typedef `base_type::point_iterator` **point_iterator**
- typedef `__rebind_v::pointer` **pointer**
- typedef `__rebind_v::reference` **reference**
- typedef `_Alloc::size_type` **size_type**
- typedef `Value_Type` **value_type**

Public Member Functions

- `iterator` **begin** ()
- `const_iterator` **begin** () const
- void **clear** ()
- bool **empty** () const
- `iterator` **end** ()
- `const_iterator` **end** () const
- void **erase** (`point_iterator`)
- template<typename Pred >
 size_type **erase_if** (Pred)
- `Cmp_Fn` & **get_cmp_fn** ()
- const `Cmp_Fn` & **get_cmp_fn** () const
- void **join** (`binomial_heap_base< Value_Type, Cmp_Fn, _Alloc >` &)
- size_type **max_size** () const
- void **modify** (`point_iterator`, const_reference)
- void **pop** ()

- `point_iterator` **push** (const_reference)
- `size_type` **size** () const
- `template<typename Pred >`
void **split** (Pred, `binomial_heap_base< Value_Type, Cmp_Fn, _Alloc >` &)
- void **swap** (`left_child_next_sibling_heap< Value_Type, Cmp_Fn, _Alloc::size_type, _Alloc >` &)
- const_reference **top** () const

Protected Types

- `typedef base_type::node` **node**
- `typedef _Alloc::template rebind< left_child_next_sibling_heap_node< Value_Type, _Alloc::size_type, _Alloc >::other` **node_allocator**
- `typedef base_type::node_const_pointer` **node_const_pointer**
- `typedef _Alloc::size_type` **node_metadata**
- `typedef base_type::node_pointer` **node_pointer**
- `typedef std::pair< node_pointer, node_pointer >` **node_pointer_pair**

Protected Member Functions

- **binomial_heap_base** (const Cmp_Fn &)
- **binomial_heap_base** (const `binomial_heap_base< Value_Type, Cmp_Fn, _Alloc >` &)
- void **actual_erase_node** (node_pointer)
- void **bubble_to_top** (node_pointer)
- void **clear_imp** (node_pointer)
- `template<typename It >`
void **copy_from_range** (It, It)
- void **find_max** ()
- node_pointer **get_new_node_for_insert** (const_reference)
- node_pointer **prune** (Pred)
- void **swap** (`binomial_heap_base< Value_Type, Cmp_Fn, _Alloc >` &)
- void **swap_with_parent** (node_pointer, node_pointer)
- void **to_linked_list** ()
- void **value_swap** (`left_child_next_sibling_heap` &)

Static Protected Member Functions

- static void **make_child_of** (node_pointer, node_pointer)
- static node_pointer **parent** (node_pointer)

Protected Attributes

- node_pointer **m_p_max**
- node_pointer **m_p_root**
- size_type **m_size**

5.225.1 Detailed Description

```
template<typename Value_Type, typename Cmp_Fn, typename _Alloc>
class __gnu_pbds::detail::binomial_heap_base< Value_Type, Cmp_Fn, _Alloc >
```

Base class for binomial heap.

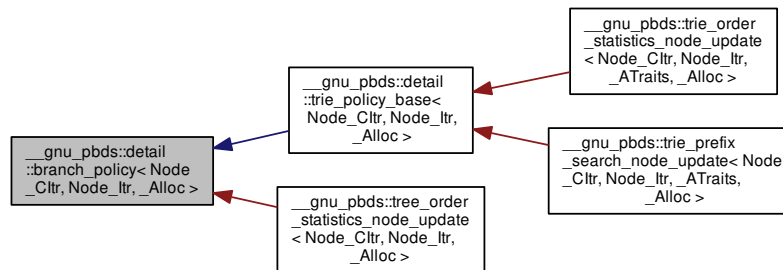
Definition at line 77 of file binomial_heap_base_.hpp.

The documentation for this class was generated from the following file:

- [binomial_heap_base_.hpp](#)

5.226 __gnu_pbds::detail::branch_policy< Node_Cltr, Node_Itr, _Alloc > Struct Template Reference

Inheritance diagram for __gnu_pbds::detail::branch_policy< Node_Cltr, Node_Itr, _Alloc >:



Protected Types

- typedef `rebind_v::const_pointer` **const_pointer**
- typedef `rebind_v::const_reference` **const_reference**
- typedef `Node_Itr::value_type` **it_type**
- typedef `rebind_k::const_reference` **key_const_reference**
- typedef `value_type::first_type` **key_type**
- typedef `remove_const< key_type >::type` **rkey_type**
- typedef `remove_const< value_type >::type` **rcvalue_type**
- typedef `_Alloc::template rebind< rkey_type >::other` **rebind_k**
- typedef `_Alloc::template rebind< rcvalue_type >::other` **rebind_v**
- typedef `rebind_v::reference` **reference**
- typedef `std::iterator_traits< it_type >::value_type` **value_type**

Protected Member Functions

- virtual `it_type` **end** ()=0
- `it_type` **end_iterator** () const

Static Protected Member Functions

- static `key_const_reference` **extract_key** (const_reference r_val)

5.226.1 Detailed Description

```
template<typename Node_Cltr, typename Node_Itr, typename _Alloc>
struct __gnu_pbds::detail::branch_policy< Node_Cltr, Node_Itr, _Alloc >
```

Primary template, base class for branch structure policies.

Definition at line 52 of file `branch_policy.hpp`.

The documentation for this struct was generated from the following file:

- [branch_policy.hpp](#)

5.227 `__gnu_pbds::detail::branch_policy< Node_Cltr, Node_Cltr, _Alloc >` Struct Template Reference

Protected Types

- typedef `rebind_v::const_pointer` **const_pointer**
- typedef `rebind_v::const_reference` **const_reference**
- typedef `Node_Cltr::value_type` **it_type**
- typedef `rebind_v::const_reference` **key_const_reference**
- typedef `value_type` **key_type**
- typedef `remove_const< value_type >::type` **rcvalue_type**
- typedef `_Alloc::template rebind< rcvalue_type >::other` **rebind_v**
- typedef `rebind_v::reference` **reference**
- typedef `std::iterator_traits< it_type >::value_type` **value_type**

Protected Member Functions

- virtual `it_type` **end** () const =0
- `it_type` **end_iterator** () const

Static Protected Member Functions

- static `key_const_reference` **extract_key** (const_reference r_val)

5.227.1 Detailed Description

```
template<typename Node_Cltr, typename _Alloc>
struct __gnu_pbds::detail::branch_policy< Node_Cltr, Node_Cltr, _Alloc >
```

Specialization for const iterators.

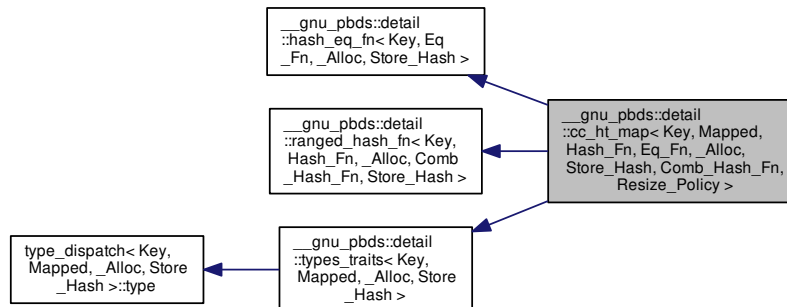
Definition at line 88 of file branch_policy.hpp.

The documentation for this struct was generated from the following file:

- [branch_policy.hpp](#)

5.228 __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy > Class Template Reference

Inheritance diagram for __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy >:



Public Types

- enum { **store_hash** }
- typedef _Alloc **allocator_type**
- typedef Comb_Hash_Fn **comb_hash_fn**
- typedef [const_iterator](#) **const_iterator**
- typedef traits_base::const_pointer **const_pointer**
- typedef traits_base::const_reference **const_reference**
- typedef _Alloc::difference_type **difference_type**
- typedef Eq_Fn **eq_fn**
- typedef Hash_Fn **hash_fn**
- typedef [iterator](#) **iterator**
- typedef traits_base::key_const_pointer **key_const_pointer**

- `typedef traits_base::key_const_reference` **key_const_reference**
- `typedef traits_base::key_pointer` **key_pointer**
- `typedef traits_base::key_reference` **key_reference**
- `typedef traits_base::key_type` **key_type**
- `typedef traits_base::mapped_const_pointer` **mapped_const_pointer**
- `typedef traits_base::mapped_const_reference` **mapped_const_reference**
- `typedef traits_base::mapped_pointer` **mapped_pointer**
- `typedef traits_base::mapped_reference` **mapped_reference**
- `typedef traits_base::mapped_type` **mapped_type**
- `typedef __nothrowcopy::indicator` **no_throw_indicator**
- `typedef point_const_iterator` **point_const_iterator**
- `typedef point_iterator` **point_iterator**
- `typedef traits_base::pointer` **pointer**
- `typedef traits_base::reference` **reference**
- `typedef Resize_Policy` **resize_policy**
- `typedef _Alloc::size_type` **size_type**
- `typedef integral_constant< int, Store_Hash >` **store_extra**
- `typedef traits_base::value_type` **value_type**

Public Member Functions

- `cc_ht_map` (const Hash_Fn &)
- `cc_ht_map` (const Hash_Fn &, const Eq_Fn &)
- `cc_ht_map` (const Hash_Fn &, const Eq_Fn &, const Comb_Hash_Fn &)
- `cc_ht_map` (const Hash_Fn &, const Eq_Fn &, const Comb_Hash_Fn &, const Resize_Policy &)
- `cc_ht_map` (const [cc_ht_map](#)< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy > &)
- iterator **begin** ()
- const_iterator **begin** () const
- void **clear** ()
- template<typename It >
void **copy_from_range** (It, It)
- bool **empty** () const
- iterator **end** ()
- const_iterator **end** () const
- bool **erase** (key_const_reference)
- template<typename Pred >
size_type **erase_if** (Pred)
- point_iterator **find** (key_const_reference)
- point_const_iterator **find** (key_const_reference) const
- point_iterator **find_end** ()
- point_const_iterator **find_end** () const
- Comb_Hash_Fn & [get_comb_hash_fn](#) ()
- const Comb_Hash_Fn & [get_comb_hash_fn](#) () const
- Eq_Fn & [get_eq_fn](#) ()
- const Eq_Fn & [get_eq_fn](#) () const
- Hash_Fn & [get_hash_fn](#) ()
- const Hash_Fn & [get_hash_fn](#) () const
- Resize_Policy & [get_resize_policy](#) ()
- const Resize_Policy & [get_resize_policy](#) () const

- void **initialize** ()
- `std::pair`< point_iterator, bool > **insert** (const_reference r_val)
- size_type **max_size** () const
- mapped_reference **operator[]** (key_const_reference r_key)
- size_type **size** () const
- void **swap** (`cc_ht_map`< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy > &)

Public Attributes

- no_throw_indicator **m_no_throw_copies_indicator**
- store_extra **m_store_extra_indicator**

Friends

- class **const_iterator_**
- class **iterator_**

5.228.1 Detailed Description

template<typename Key, typename Mapped, typename Hash_Fn, typename Eq_Fn, typename _Alloc, bool Store_Hash, typename Comb_Hash_Fn, typename Resize_Policy>

class `__gnu_pbds::detail::cc_ht_map`< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy >

A collision-chaining hash-based container.

Template Parameters

<i>Key</i>	Key type.
<i>Mapped</i>	Map type.
<i>Hash_Fn</i>	Hashing functor. Default is <code>__gnu_cxx::hash</code> .
<i>Eq_Fn</i>	Equal functor. Default <code>std::equal_to<Key></code>
<i>_Alloc</i>	Allocator type.
<i>Store_Hash</i>	If key type stores extra metadata. Defaults to false.
<i>Comb_Hash_Fn</i>	Combining hash functor. If <i>Hash_Fn</i> is not null_type, then this is the ranged-hash functor; otherwise, this is the range-hashing functor. XXX(See Design::Hash-Based Containers::Hash Policies.) Default <code>direct_mask_range_hashing</code> .
<i>Resize_Policy</i>	Resizes hash. Defaults to <code>hash_standard_resize_policy</code> , using <code>hash_exponential_size_policy</code> and <code>hash_load_check_resize_trigger</code> .

Bases are: `detail::hash_eq_fn`, `Resize_Policy`, `detail::ranged_hash_fn`, `detail::types_traits`. (Optional: `detail::debug_map_base`.)

Definition at line 139 of file `cc_ht_map.hpp`.

5.228.2 Member Enumeration Documentation

5.228.2.1 `template<typename Key, typename Mapped, typename Hash_Fn, typename Eq_Fn, typename _Alloc, bool Store_Hash, typename Comb_Hash_Fn, typename Resize_Policy> anonymous enum`

Value stores hash, true or false.

Definition at line 200 of file `cc_ht_map.hpp`.

5.228.3 Member Function Documentation

5.228.3.1 `template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc , bool Store_Hash, typename Comb_Hash_Fn , typename Resize_Policy > bool __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy >::empty () const [inline]`

True if `size() == 0`.

Definition at line 52 of file `cc_ht_map.hpp`.

5.228.3.2 `template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc , bool Store_Hash, typename Comb_Hash_Fn , typename Resize_Policy > Comb_Hash_Fn & __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy >::get_comb_hash_fn ()`

Return current `comb_hash_fn`.

Definition at line 70 of file `cc_ht_map.hpp`.

5.228.3.3 `template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc , bool Store_Hash, typename Comb_Hash_Fn , typename Resize_Policy > const Comb_Hash_Fn & __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy >::get_comb_hash_fn () const`

Return current `const comb_hash_fn`.

Definition at line 76 of file `cc_ht_map.hpp`.

5.228.3.4 `template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc , bool Store_Hash, typename Comb_Hash_Fn , typename Resize_Policy > Eq_Fn & __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy >::get_eq_fn ()`

Return current `eq_fn`.

Definition at line 58 of file `cc_ht_map.hpp`.

5.228.3.5 `template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc , bool Store_Hash, typename Comb_Hash_Fn , typename Resize_Policy > const Eq_Fn & __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy >::get_eq_fn () const`

Return current `const eq_fn`.

Definition at line 64 of file `cc_ht_map.hpp`.

5.228.3.6 `template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc , bool Store_Hash, typename Comb_Hash_Fn , typename Resize_Policy > Hash_Fn & __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy >::get_hash_fn ()`

Return current hash_fn.

Definition at line 46 of file cc_ht_map_.hpp.

5.228.3.7 `template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc , bool Store_Hash, typename Comb_Hash_Fn , typename Resize_Policy > const Hash_Fn & __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy >::get_hash_fn () const`

Return current const hash_fn.

Definition at line 52 of file cc_ht_map_.hpp.

5.228.3.8 `template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc , bool Store_Hash, typename Comb_Hash_Fn , typename Resize_Policy > Resize_Policy & __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy >::get_resize_policy ()`

Return current resize_policy.

Definition at line 82 of file cc_ht_map_.hpp.

5.228.3.9 `template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc , bool Store_Hash, typename Comb_Hash_Fn , typename Resize_Policy > const Resize_Policy & __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy >::get_resize_policy () const`

Return current const resize_policy.

Definition at line 88 of file cc_ht_map_.hpp.

The documentation for this class was generated from the following file:

- [cc_ht_map_.hpp](#)

5.229 `__gnu_pbds::detail::cond_dealtor< Entry, _Alloc >` Class Template Reference

Public Types

- `typedef HT_Map::entry entry`
- `typedef HT_Map::entry_allocator entry_allocator`
- `typedef __rebind_e::other entry_allocator`
- `typedef entry_allocator::pointer entry_pointer`
- `typedef HT_Map::key_type key_type`

Public Member Functions

- **cond_dealtor** (entry_allocator *p_a, entry *p_e)
- **cond_dealtor** (entry_pointer p_e)
- void **set_key_destruct** ()
- void **set_no_action** ()
- void **set_no_action_destructor** ()

Protected Attributes

- bool **m_key_destruct**
- entry_allocator *const **m_p_a**
- entry *const **m_p_e**

5.229.1 Detailed Description

```
template<typename Entry, typename _Alloc>
class __gnu_pbds::detail::cond_dealtor< Entry, _Alloc >
```

Conditional deallocate constructor argument.

Conditional key destructor, cc_hash.

Definition at line 50 of file cond_dealtor.hpp.

The documentation for this class was generated from the following files:

- [cond_dealtor.hpp](#)
- [cond_key_dtor_entry_dealtor.hpp](#)

5.230 `__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, Tag, Policy_TI >` Struct Template Reference

5.230.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc, typename Tag, typename Policy_TI = null_type>
struct __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, Tag, Policy_TI >
```

Dispatch mechanism, primary template for associative types.

Definition at line 449 of file tag_and_trait.hpp.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.231 `__gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, binary_heap_tag, null_type >` Struct Template Reference

Public Types

- typedef [binary_heap](#)<_VTp, Cmp_Fn, _Alloc > [type](#)

5.231.1 Detailed Description

```
template<typename _VTp, typename Cmp_Fn, typename _Alloc>
struct __gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, binary_heap_tag, null_type >
```

Specialization for `binary_heap`.

Definition at line 95 of file `priority_queue_base_dispatch.hpp`.

5.231.2 Member Typedef Documentation

5.231.2.1 `template<typename _VTp, typename Cmp_Fn, typename _Alloc > typedef binary_heap<_VTp, Cmp_Fn, _Alloc> __gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, binary_heap_tag, null_type >::type`

Dispatched type.

Definition at line 99 of file `priority_queue_base_dispatch.hpp`.

The documentation for this struct was generated from the following file:

- [priority_queue_base_dispatch.hpp](#)

5.232 `__gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, binomial_heap_tag, null_type >` Struct Template Reference

Public Types

- typedef [binomial_heap](#)<_VTp, Cmp_Fn, _Alloc > [type](#)

5.232.1 Detailed Description

```
template<typename _VTp, typename Cmp_Fn, typename _Alloc>
struct __gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, binomial_heap_tag, null_type >
```

Specialization for `binomial_heap`.

Definition at line 77 of file `priority_queue_base_dispatch.hpp`.

5.232.2 Member Typedef Documentation

5.232.2.1 `template<typename _VTp, typename Cmp_Fn, typename _Alloc> typedef binomial_heap<_VTp, Cmp_Fn, _Alloc>
__gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, binomial_heap_tag, null_type
>::type`

Dispatched type.

Definition at line 81 of file `priority_queue_base_dispatch.hpp`.

The documentation for this struct was generated from the following file:

- [priority_queue_base_dispatch.hpp](#)

5.233 `__gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, pairing_heap_tag, null_type>` Struct Template Reference

Public Types

- typedef `pairing_heap<_VTp, Cmp_Fn, _Alloc>` `type`

5.233.1 Detailed Description

```
template<typename _VTp, typename Cmp_Fn, typename _Alloc>
struct __gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, pairing_heap_tag, null_type>
```

Specialization for `pairing_heap`.

Definition at line 68 of file `priority_queue_base_dispatch.hpp`.

5.233.2 Member Typedef Documentation

5.233.2.1 `template<typename _VTp, typename Cmp_Fn, typename _Alloc> typedef pairing_heap<_VTp, Cmp_Fn, _Alloc>
__gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, pairing_heap_tag, null_type
>::type`

Dispatched type.

Definition at line 72 of file `priority_queue_base_dispatch.hpp`.

The documentation for this struct was generated from the following file:

- [priority_queue_base_dispatch.hpp](#)

5.234 `__gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, rc_binomial_heap_tag, null_type>` > Struct Template Reference

Public Types

- typedef [rc_binomial_heap](#)<_VTp, Cmp_Fn, _Alloc> [type](#)

5.234.1 Detailed Description

```
template<typename _VTp, typename Cmp_Fn, typename _Alloc>
struct __gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, rc_binomial_heap_tag, null_type>
```

Specialization for rc_binary_heap.

Definition at line 86 of file priority_queue_base_dispatch.hpp.

5.234.2 Member Typedef Documentation

5.234.2.1 `template<typename _VTp, typename Cmp_Fn, typename _Alloc> typedef rc_binomial_heap<_VTp, Cmp_Fn, _Alloc> __gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, rc_binomial_heap_tag, null_type>::type`

Dispatched type.

Definition at line 90 of file priority_queue_base_dispatch.hpp.

The documentation for this struct was generated from the following file:

- [priority_queue_base_dispatch.hpp](#)

5.235 `__gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, thin_heap_tag, null_type>` > Struct Template Reference

Public Types

- typedef [thin_heap](#)<_VTp, Cmp_Fn, _Alloc> [type](#)

5.235.1 Detailed Description

```
template<typename _VTp, typename Cmp_Fn, typename _Alloc>
struct __gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, thin_heap_tag, null_type>
```

Specialization for thin_heap.

Definition at line 104 of file priority_queue_base_dispatch.hpp.

5.235.2 Member Typedef Documentation

5.235.2.1 `template<typename VTp, typename Cmp_Fn, typename _Alloc > typedef thin_heap<VTp, Cmp_Fn, _Alloc> __gnu_pbds::detail::container_base_dispatch<VTp, Cmp_Fn, _Alloc, thin_heap_tag, null_type>::type`

Dispatched type.

Definition at line 108 of file `priority_queue_base_dispatch.hpp`.

The documentation for this struct was generated from the following file:

- [priority_queue_base_dispatch.hpp](#)

5.236 `__gnu_pbds::detail::container_base_dispatch`< Key, Mapped, _Alloc, cc_hash_tag, Policy_TI > Struct Template Reference

Public Types

- typedef `cc_ht_map`< Key, Mapped, at0t, at1t, _Alloc, at3t::value, at4t, at2t > [type](#)

5.236.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc, typename Policy_TI>
struct __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, cc_hash_tag, Policy_TI >
```

Specialization collision-chaining hash map.

Definition at line 258 of file `container_base_dispatch.hpp`.

5.236.2 Member Typedef Documentation

5.236.2.1 `template<typename Key, typename Mapped, typename _Alloc, typename Policy_TI > typedef cc_ht_map<Key, Mapped, at0t, at1t, _Alloc, at3t::value, at4t, at2t> __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, cc_hash_tag, Policy_TI >::type`

Dispatched type.

Definition at line 275 of file `container_base_dispatch.hpp`.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

5.237 `__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, gp_hash_tag, Policy_Tl >` Struct Template Reference

Public Types

- typedef [gp_ht_map](#)< Key, Mapped, at0t, at1t, _Alloc, at3t::value, at4t, at5t, at2t > [type](#)

5.237.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc, typename Policy_Tl>
struct __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, gp_hash_tag, Policy_Tl >
```

Specialization general-probe hash map.

Definition at line 303 of file `container_base_dispatch.hpp`.

5.237.2 Member Typedef Documentation

5.237.2.1 `template<typename Key , typename Mapped , typename _Alloc , typename Policy_Tl > typedef gp_ht_map<Key, Mapped, at0t, at1t, _Alloc, at3t::value, at4t, at5t, at2t> __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, gp_hash_tag, Policy_Tl >::type`

Dispatched type.

Definition at line 322 of file `container_base_dispatch.hpp`.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

5.238 `__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, list_update_tag, Policy_Tl >` Struct Template Reference

Public Types

- typedef [lu_map](#)< Key, Mapped, at0t, _Alloc, at1t > [type](#)

5.238.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc, typename Policy_Tl>
struct __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, list_update_tag, Policy_Tl >
```

Specialization for list-update map.

Definition at line 107 of file `container_base_dispatch.hpp`.

5.238.2 Member Typedef Documentation

5.238.2.1 `template<typename Key , typename Mapped , typename _Alloc , typename Policy_Tl > typedef lu_map<Key, Mapped, at0t, _Alloc, at1t> __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, list_update_tag, Policy_Tl >::type`

Dispatched type.

Definition at line 118 of file `container_base_dispatch.hpp`.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

5.239 `__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, ov_tree_tag, Policy_Tl >` Struct Template Reference

Public Types

- typedef `ov_tree_map< Key, Mapped, at0t, at1t, _Alloc >` `type`

5.239.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc, typename Policy_Tl>
struct __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, ov_tree_tag, Policy_Tl >
```

Specialization ordered-vector tree map.

Definition at line 227 of file `container_base_dispatch.hpp`.

5.239.2 Member Typedef Documentation

5.239.2.1 `template<typename Key , typename Mapped , typename _Alloc , typename Policy_Tl > typedef ov_tree_map<Key, Mapped, at0t, at1t, _Alloc> __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, ov_tree_tag, Policy_Tl >::type`

Dispatched type.

Definition at line 237 of file `container_base_dispatch.hpp`.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

5.240 `__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, pat_trie_tag, Policy_Tl >` Struct Template Reference

Public Types

- typedef `pat_trie_map< Key, Mapped, at1t, _Alloc >` **type**

5.240.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc, typename Policy_Tl>
struct __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, pat_trie_tag, Policy_Tl >
```

Specialization for PATRICIA trie map.

Definition at line 139 of file `container_base_dispatch.hpp`.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

5.241 `__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, rb_tree_tag, Policy_Tl >` Struct Template Reference

Public Types

- typedef `rb_tree_map< Key, Mapped, at0t, at1t, _Alloc >` **type**

5.241.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc, typename Policy_Tl>
struct __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, rb_tree_tag, Policy_Tl >
```

Specialization for R-B tree map.

Definition at line 165 of file `container_base_dispatch.hpp`.

5.241.2 Member Typedef Documentation

5.241.2.1 `template<typename Key , typename Mapped , typename _Alloc , typename Policy_Tl > typedef rb_tree_map<Key, Mapped, at0t, at1t, _Alloc> __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, rb_tree_tag, Policy_Tl >::type`

Dispatched type.

Definition at line 175 of file `container_base_dispatch.hpp`.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

5.242 `__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, splay_tree_tag, Policy_Tl >` Struct Template Reference

Public Types

- typedef `splay_tree_map< Key, Mapped, at0t, at1t, _Alloc >` `type`

5.242.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc, typename Policy_Tl>
struct __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, splay_tree_tag, Policy_Tl >
```

Specialization splay tree map.

Definition at line 195 of file `container_base_dispatch.hpp`.

5.242.2 Member Typedef Documentation

5.242.2.1 `template<typename Key , typename Mapped , typename _Alloc , typename Policy_Tl > typedef splay_tree_map<Key, Mapped, at0t, at1t, _Alloc> __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, splay_tree_tag, Policy_Tl >::type`

Dispatched type.

Definition at line 206 of file `container_base_dispatch.hpp`.

The documentation for this struct was generated from the following file:

- `container_base_dispatch.hpp`

5.243 `__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, cc_hash_tag, Policy_Tl >` Struct Template Reference

Public Types

- typedef `cc_ht_set< Key, null_type, at0t, at1t, _Alloc, at3t::value, at4t, at2t >` `type`

5.243.1 Detailed Description

```
template<typename Key, typename _Alloc, typename Policy_Tl>
struct __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, cc_hash_tag, Policy_Tl >
```

Specialization collision-chaining hash set.

Definition at line 280 of file `container_base_dispatch.hpp`.

5.243.2 Member Typedef Documentation

5.243.2.1 `template<typename Key , typename _Alloc , typename Policy_TI > typedef cc_ht_set<Key, null_type, at0t, at1t, _Alloc, at3t::value, at4t, at2t> __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, cc_hash_tag, Policy_TI >::type`

Dispatched type.

Definition at line 298 of file `container_base_dispatch.hpp`.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

5.244 __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, gp_hash_tag, Policy_TI > Struct Template Reference

Public Types

- `typedef gp_ht_set< Key, null_type, at0t, at1t, _Alloc, at3t::value, at4t, at5t, at2t > type`

5.244.1 Detailed Description

```
template<typename Key, typename _Alloc, typename Policy_TI>
struct __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, gp_hash_tag, Policy_TI >
```

Specialization general-probe hash set.

Definition at line 327 of file `container_base_dispatch.hpp`.

5.244.2 Member Typedef Documentation

5.244.2.1 `template<typename Key , typename _Alloc , typename Policy_TI > typedef gp_ht_set<Key, null_type, at0t, at1t, _Alloc, at3t::value, at4t, at5t, at2t> __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, gp_hash_tag, Policy_TI >::type`

Dispatched type.

Definition at line 347 of file `container_base_dispatch.hpp`.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

5.245 `__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, list_update_tag, Policy_TI >` Struct Template Reference

Public Types

- `typedef lu_set< Key, null_type, at0t, _Alloc, at1t > type`

5.245.1 Detailed Description

```
template<typename Key, typename _Alloc, typename Policy_TI>
struct __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, list_update_tag, Policy_TI >
```

Specialization for list-update set.

Definition at line 123 of file `container_base_dispatch.hpp`.

5.245.2 Member Typedef Documentation

5.245.2.1 `template<typename Key , typename _Alloc , typename Policy_TI > typedef lu_set<Key, null_type, at0t, _Alloc, at1t> __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, list_update_tag, Policy_TI >::type`

Dispatched type.

Definition at line 134 of file `container_base_dispatch.hpp`.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

5.246 `__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, ov_tree_tag, Policy_TI >` Struct Template Reference

Public Types

- `typedef ov_tree_set< Key, null_type, at0t, at1t, _Alloc > type`

5.246.1 Detailed Description

```
template<typename Key, typename _Alloc, typename Policy_TI>
struct __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, ov_tree_tag, Policy_TI >
```

Specialization ordered-vector tree set.

Definition at line 242 of file `container_base_dispatch.hpp`.

5.246.2 Member Typedef Documentation

5.246.2.1 `template<typename Key , typename _Alloc , typename Policy_TI > typedef ov_tree_set<Key, null_type, at0t, at1t, _Alloc> __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, ov_tree_tag, Policy_TI >::type`

Dispatched type.

Definition at line 253 of file `container_base_dispatch.hpp`.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

5.247 __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, pat_trie_tag, Policy_TI > Struct Template Reference

Public Types

- `typedef pat_trie_set< Key, null_type, at1t, _Alloc > type`

5.247.1 Detailed Description

```
template<typename Key, typename _Alloc, typename Policy_TI>
struct __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, pat_trie_tag, Policy_TI >
```

Specialization for PATRICIA trie set.

Definition at line 151 of file `container_base_dispatch.hpp`.

5.247.2 Member Typedef Documentation

5.247.2.1 `template<typename Key , typename _Alloc , typename Policy_TI > typedef pat_trie_set<Key, null_type, at1t, _Alloc> __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, pat_trie_tag, Policy_TI >::type`

Dispatched type.

Definition at line 160 of file `container_base_dispatch.hpp`.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

5.248 `__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, rb_tree_tag, Policy_Tl >` Struct Template Reference

Public Types

- `typedef rb_tree_set< Key, null_type, at0t, at1t, _Alloc >` **type**

5.248.1 Detailed Description

```
template<typename Key, typename _Alloc, typename Policy_Tl>
struct __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, rb_tree_tag, Policy_Tl >
```

Specialization for R-B tree set.

Definition at line 180 of file `container_base_dispatch.hpp`.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

5.249 `__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, splay_tree_tag, Policy_Tl >` Struct Template Reference

Public Types

- `typedef splay_tree_set< Key, null_type, at0t, at1t, _Alloc >` [type](#)

5.249.1 Detailed Description

```
template<typename Key, typename _Alloc, typename Policy_Tl>
struct __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, splay_tree_tag, Policy_Tl >
```

Specialization splay tree set.

Definition at line 211 of file `container_base_dispatch.hpp`.

5.249.2 Member Typedef Documentation

5.249.2.1 `template<typename Key, typename _Alloc, typename Policy_Tl > typedef splay_tree_set<Key, null_type, at0t, at1t, _Alloc> __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, splay_tree_tag, Policy_Tl >::type`

Dispatched type.

Definition at line 222 of file `container_base_dispatch.hpp`.

The documentation for this struct was generated from the following file:

- [container_base_dispatch.hpp](#)

5.250 `__gnu_pbds::detail::default_comb_hash_fn` Struct Reference

Public Types

- typedef [direct_mask_range_hashing](#) type

5.250.1 Detailed Description

Primary template, `default_comb_hash_fn`.

Definition at line 80 of file `standard_policies.hpp`.

5.250.2 Member Typedef Documentation

5.250.2.1 `typedef direct_mask_range_hashing __gnu_pbds::detail::default_comb_hash_fn::type`

Dispatched type.

Definition at line 83 of file `standard_policies.hpp`.

The documentation for this struct was generated from the following file:

- [standard_policies.hpp](#)

5.251 `__gnu_pbds::detail::default_eq_fn< Key >` Struct Template Reference

Public Types

- typedef [std::equal_to< Key >](#) type

5.251.1 Detailed Description

```
template<typename Key>
struct __gnu_pbds::detail::default_eq_fn< Key >
```

Primary template, `default_eq_fn`.

Definition at line 67 of file `standard_policies.hpp`.

5.251.2 Member Typedef Documentation

5.251.2.1 `template<typename Key> typedef std::equal_to<Key> __gnu_pbds::detail::default_eq_fn< Key >::type`

Dispatched type.

Definition at line 70 of file `standard_policies.hpp`.

The documentation for this struct was generated from the following file:

- [standard_policies.hpp](#)

5.252 `__gnu_pbds::detail::default_hash_fn< Key >` Struct Template Reference

Public Types

- `typedef std::tr1::hash< Key > type`

5.252.1 Detailed Description

```
template<typename Key>
struct __gnu_pbds::detail::default_hash_fn< Key >
```

Primary template, `default_hash_fn`.

Definition at line 59 of file `standard_policies.hpp`.

5.252.2 Member Typedef Documentation

5.252.2.1 `template<typename Key> typedef std::tr1::hash<Key> __gnu_pbds::detail::default_hash_fn< Key >::type`

Dispatched type.

Definition at line 62 of file `standard_policies.hpp`.

The documentation for this struct was generated from the following file:

- [standard_policies.hpp](#)

5.253 `__gnu_pbds::detail::default_probe_fn< Comb_Probe_Fn >` Struct Template Reference

Public Types

- `typedef cond_type::__type type`

5.253.1 Detailed Description

```
template<typename Comb_Probe_Fn>
struct __gnu_pbds::detail::default_probe_fn< Comb_Probe_Fn >
```

Primary template, default_probe_fn.

Definition at line 117 of file standard_policies.hpp.

5.253.2 Member Typedef Documentation

5.253.2.1 `template<typename Comb_Probe_Fn> typedef cond_type::__type __gnu_pbds::detail::default_probe_fn< Comb_Probe_Fn >::type`

Dispatched type.

Definition at line 129 of file standard_policies.hpp.

The documentation for this struct was generated from the following file:

- [standard_policies.hpp](#)

5.254 __gnu_pbds::detail::default_resize_policy< Comb_Hash_Fn > Struct Template Reference

Public Types

- typedef [hash_standard_resize_policy](#)< size_policy_type, [trigger](#), false, size_type > [type](#)

5.254.1 Detailed Description

```
template<typename Comb_Hash_Fn>
struct __gnu_pbds::detail::default_resize_policy< Comb_Hash_Fn >
```

Primary template, default_resize_policy.

Definition at line 88 of file standard_policies.hpp.

5.254.2 Member Typedef Documentation

5.254.2.1 `template<typename Comb_Hash_Fn> typedef hash_standard_resize_policy<size_policy_type, trigger, false, size_type> __gnu_pbds::detail::default_resize_policy< Comb_Hash_Fn >::type`

Dispatched type.

Definition at line 105 of file standard_policies.hpp.

The documentation for this struct was generated from the following file:

- [standard_policies.hpp](#)

5.255 `__gnu_pbds::detail::default_trie_access_traits< Key >` Struct Template Reference

5.255.1 Detailed Description

```
template<typename Key>
struct __gnu_pbds::detail::default_trie_access_traits< Key >
```

Primary template, `default_trie_access_traits`.

Definition at line 135 of file `standard_policies.hpp`.

The documentation for this struct was generated from the following file:

- [standard_policies.hpp](#)

5.256 `__gnu_pbds::detail::default_trie_access_traits< std::basic_string< Char, Char_Traits, std::allocator< char > > >` Struct Template Reference

Public Types

- typedef [trie_string_access_traits< string_type >](#) `type`

5.256.1 Detailed Description

```
template<typename Char, typename Char_Traits>
struct __gnu_pbds::detail::default_trie_access_traits< std::basic_string< Char, Char_Traits, std::allocator< char > > >
```

Partial specialization, `default_trie_access_traits`.

Definition at line 142 of file `standard_policies.hpp`.

5.256.2 Member Typedef Documentation

5.256.2.1 `template<typename Char , typename Char_Traits > typedef trie_string_access_traits<string_type> __gnu_pbds::detail::default_trie_access_traits< std::basic_string< Char, Char_Traits, std::allocator< char > > >::type`

Dispatched type.

Definition at line 149 of file `standard_policies.hpp`.

The documentation for this struct was generated from the following file:

- [standard_policies.hpp](#)

5.257 `__gnu_pbds::detail::default_update_policy` Struct Reference

Public Types

- typedef [lu_move_to_front_policy](#) type

5.257.1 Detailed Description

Default update policy.

Definition at line 109 of file `standard_policies.hpp`.

5.257.2 Member Typedef Documentation

5.257.2.1 typedef `lu_move_to_front_policy` `__gnu_pbds::detail::default_update_policy::type`

Dispatched type.

Definition at line 112 of file `standard_policies.hpp`.

The documentation for this struct was generated from the following file:

- [standard_policies.hpp](#)

5.258 `__gnu_pbds::detail::dumnode_const_iterator< Key, Data, _Alloc >` Struct Template Reference

Public Types

- typedef `const_iterator` **const_reference**
- typedef `const_reference` **reference**
- typedef `const_iterator` **value_type**

5.258.1 Detailed Description

```
template<typename Key, typename Data, typename _Alloc>
struct __gnu_pbds::detail::dumnode_const_iterator< Key, Data, _Alloc >
```

Constant node iterator.

Definition at line 52 of file `null_node_metadata.hpp`.

The documentation for this struct was generated from the following file:

- [null_node_metadata.hpp](#)

5.259 `__gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, No_Throw>` Struct Template Reference

5.259.1 Detailed Description

```
template<typename _VTp, typename Cmp_Fn, typename _Alloc, bool No_Throw>
struct __gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, No_Throw>
```

Entry compare, primary template.

Definition at line 50 of file `entry_cmp.hpp`.

The documentation for this struct was generated from the following file:

- [entry_cmp.hpp](#)

5.260 `__gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, false>` Struct Template Reference

Classes

- struct [type](#)

Public Types

- typedef `__rebind_v::other::const_pointer` **entry**

5.260.1 Detailed Description

```
template<typename _VTp, typename Cmp_Fn, typename _Alloc>
struct __gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, false>
```

Specialization, false.

Definition at line 62 of file `entry_cmp.hpp`.

The documentation for this struct was generated from the following file:

- [entry_cmp.hpp](#)

5.261 `__gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, false>::type` Struct Reference

Inherits `Cmp_Fn`.

Public Member Functions

- **type** (const Cmp_Fn &other)
- bool **operator()** (entry lhs, entry rhs) const

5.261.1 Detailed Description

```
template<typename _VTp, typename Cmp_Fn, typename _Alloc>
struct __gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, false >::type
```

Compare plus entry.

Definition at line 71 of file entry_cmp.hpp.

The documentation for this struct was generated from the following file:

- [entry_cmp.hpp](#)

5.262 __gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, true > Struct Template Reference

Public Types

- typedef Cmp_Fn [type](#)

5.262.1 Detailed Description

```
template<typename _VTp, typename Cmp_Fn, typename _Alloc>
struct __gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, true >
```

Specialization, true.

Definition at line 54 of file entry_cmp.hpp.

5.262.2 Member Typedef Documentation

```
5.262.2.1 template<typename _VTp , typename Cmp_Fn , typename _Alloc > typedef Cmp_Fn
__gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, true >::type
```

Compare.

Definition at line 57 of file entry_cmp.hpp.

The documentation for this struct was generated from the following file:

- [entry_cmp.hpp](#)

5.263 `__gnu_pbds::detail::entry_pred<_VTp, Pred, _Alloc, No_Throw >` Struct Template Reference

5.263.1 Detailed Description

```
template<typename _VTp, typename Pred, typename _Alloc, bool No_Throw>
struct __gnu_pbds::detail::entry_pred<_VTp, Pred, _Alloc, No_Throw >
```

Entry predicate primary class template.

Definition at line 50 of file `entry_pred.hpp`.

The documentation for this struct was generated from the following file:

- [entry_pred.hpp](#)

5.264 `__gnu_pbds::detail::entry_pred<_VTp, Pred, _Alloc, false >` Struct Template Reference

Public Types

- typedef `__rebind_v::other::const_pointer` **entry**

5.264.1 Detailed Description

```
template<typename _VTp, typename Pred, typename _Alloc>
struct __gnu_pbds::detail::entry_pred<_VTp, Pred, _Alloc, false >
```

Specialization, `false`.

Definition at line 61 of file `entry_pred.hpp`.

The documentation for this struct was generated from the following file:

- [entry_pred.hpp](#)

5.265 `__gnu_pbds::detail::entry_pred<_VTp, Pred, _Alloc, true >` Struct Template Reference

Public Types

- typedef `Pred` **type**

5.265.1 Detailed Description

```
template<typename _VTp, typename Pred, typename _Alloc>
struct __gnu_pbds::detail::entry_pred< _VTp, Pred, _Alloc, true >
```

Specialization, true.

Definition at line 54 of file entry_pred.hpp.

The documentation for this struct was generated from the following file:

- [entry_pred.hpp](#)

5.266 __gnu_pbds::detail::eq_by_less< Key, Cmp_Fn > Struct Template Reference

Inherits Cmp_Fn.

Public Member Functions

- bool **operator()** (const Key &r_lhs, const Key &r_rhs) const

5.266.1 Detailed Description

```
template<typename Key, class Cmp_Fn>
struct __gnu_pbds::detail::eq_by_less< Key, Cmp_Fn >
```

Equivalence function.

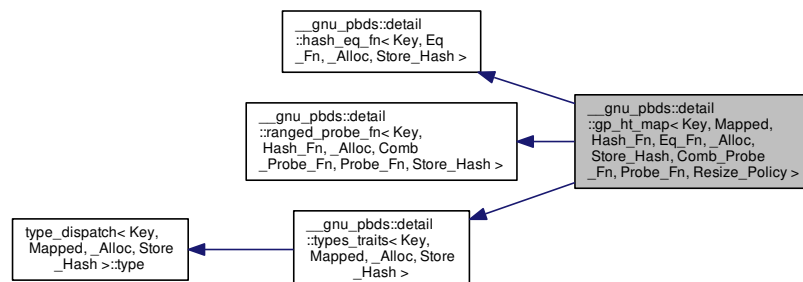
Definition at line 56 of file eq_by_less.hpp.

The documentation for this struct was generated from the following file:

- [eq_by_less.hpp](#)

5.267 __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy > Class Template Reference

Inheritance diagram for __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >:



Public Types

- enum { **store_hash** }
- typedef `_Alloc` **allocator_type**
- typedef `Comb_Probe_Fn` **comb_probe_fn**
- typedef `const_iterator` **const_iterator**
- typedef `traits_base::const_pointer` **const_pointer**
- typedef `traits_base::const_reference` **const_reference**
- typedef `_Alloc::difference_type` **difference_type**
- typedef `Eq_Fn` **eq_fn**
- typedef `Hash_Fn` **hash_fn**
- typedef `iterator` **iterator**
- typedef `traits_base::key_const_pointer` **key_const_pointer**
- typedef `traits_base::key_const_reference` **key_const_reference**
- typedef `traits_base::key_pointer` **key_pointer**
- typedef `traits_base::key_reference` **key_reference**
- typedef `traits_base::key_type` **key_type**
- typedef `traits_base::mapped_const_pointer` **mapped_const_pointer**
- typedef `traits_base::mapped_const_reference` **mapped_const_reference**
- typedef `traits_base::mapped_pointer` **mapped_pointer**
- typedef `traits_base::mapped_reference` **mapped_reference**
- typedef `traits_base::mapped_type` **mapped_type**
- typedef `__nothrowcopy::indicator` **no_throw_indicator**
- typedef `point_const_iterator` **point_const_iterator**
- typedef `point_iterator` **point_iterator**
- typedef `traits_base::pointer` **pointer**
- typedef `Probe_Fn` **probe_fn**
- typedef `traits_base::reference` **reference**
- typedef `Resize_Policy` **resize_policy**
- typedef `_Alloc::size_type` **size_type**
- typedef `integral_constant< int, Store_Hash >` **store_extra**
- typedef `traits_base::value_type` **value_type**

Public Member Functions

- **gp_ht_map** (const `gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >` &)
- **gp_ht_map** (const `Hash_Fn` &)
- **gp_ht_map** (const `Hash_Fn` &, const `Eq_Fn` &)
- **gp_ht_map** (const `Hash_Fn` &, const `Eq_Fn` &, const `Comb_Probe_Fn` &)
- **gp_ht_map** (const `Hash_Fn` &, const `Eq_Fn` &, const `Comb_Probe_Fn` &, const `Probe_Fn` &)
- **gp_ht_map** (const `Hash_Fn` &, const `Eq_Fn` &, const `Comb_Probe_Fn` &, const `Probe_Fn` &, const `Resize_Policy` &)
- iterator **begin** ()
- const_iterator **begin** () const
- void **clear** ()
- template<typename It >
void **copy_from_range** (It, It)
- bool **empty** () const
- iterator **end** ()

- `const_iterator` **end** () const
- `bool` **erase** (key_const_reference)
- `template<typename Pred >`
`size_type` **erase_if** (Pred)
- `point_iterator` **find** (key_const_reference)
- `point_const_iterator` **find** (key_const_reference) const
- `point_iterator` **find_end** ()
- `point_const_iterator` **find_end** () const
- `Comb_Probe_Fn` & [get_comb_probe_fn](#) ()
- `const Comb_Probe_Fn` & [get_comb_probe_fn](#) () const
- `Eq_Fn` & [get_eq_fn](#) ()
- `const Eq_Fn` & [get_eq_fn](#) () const
- `Hash_Fn` & [get_hash_fn](#) ()
- `const Hash_Fn` & [get_hash_fn](#) () const
- `Probe_Fn` & [get_probe_fn](#) ()
- `const Probe_Fn` & [get_probe_fn](#) () const
- `Resize_Policy` & [get_resize_policy](#) ()
- `const Resize_Policy` & [get_resize_policy](#) () const
- `std::pair< point_iterator, bool >` **insert** (const_reference r_val)
- `size_type` **max_size** () const
- `mapped_reference` **operator[]** (key_const_reference r_key)
- `size_type` **size** () const
- `void` **swap** ([gp_ht_map](#)< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy > &)

Public Attributes

- `no_throw_indicator` **m_no_throw_copies_indicator**
- `store_extra` **m_store_extra_indicator**

Friends

- `class` **const_iterator_**
- `class` **iterator_**

5.267.1 Detailed Description

```
template<typename Key, typename Mapped, typename Hash_Fn, typename Eq_Fn, typename _Alloc, bool Store_Hash, typename
Comb_Probe_Fn, typename Probe_Fn, typename Resize_Policy>
class __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy
>
```

A general-probing hash-based container.

Template Parameters

<i>Key</i>	Key type.
<i>Mapped</i>	Map type.

Template Parameters

<i>Hash_Fn</i>	Hashing functor. Default is <code>__gnu_cxx::hash</code> .
<i>Eq_Fn</i>	Equal functor. Default <code>std::equal_to<Key></code>
<i>_Alloc</i>	Allocator type.
<i>Store_Hash</i>	If key type stores extra metadata. Defaults to false.
<i>Comb_Probe_Fn</i>	Combining probe functor. If <i>Hash_Fn</i> is not null_type, then this is the ranged-probe functor; otherwise, this is the range-hashing functor. XXX See Design::Hash-Based Containers::Hash Policies. Default <code>direct_mask_range_hashing</code> .
<i>Probe_Fn</i>	Probe functor. Defaults to <code>linear_probe_fn</code> , also <code>quadratic_probe_fn</code> .
<i>Resize_Policy</i>	Resizes hash. Defaults to <code>hash_standard_resize_policy</code> , using <code>hash_exponential_size_policy</code> and <code>hash_load_check_resize_trigger</code> .

Bases are: `detail::hash_eq_fn`, `Resize_Policy`, `detail::ranged_probe_fn`, `detail::types_traits`. (Optional: `detail::debug_map_base`.)

Definition at line 142 of file `gp_ht_map.hpp`.

5.267.2 Member Enumeration Documentation

5.267.2.1 `template<typename Key, typename Mapped, typename Hash_Fn, typename Eq_Fn, typename _Alloc, bool Store_Hash, typename Comb_Probe_Fn, typename Probe_Fn, typename Resize_Policy>` anonymous enum

Value stores hash, true or false.

Definition at line 208 of file `gp_ht_map.hpp`.

5.267.3 Member Function Documentation

5.267.3.1 `template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc , bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy > bool __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >::empty () const` `[inline]`

True if `size() == 0`.

Definition at line 58 of file `gp_ht_map.hpp`.

5.267.3.2 `template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc , bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy > Comb_Probe_Fn & __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >::get_comb_probe_fn ()`

Return current `comb_probe_fn`.

Definition at line 82 of file `gp_ht_map.hpp`.

5.267.3.3 `template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc , bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy > const Comb_Probe_Fn & __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >::get_comb_probe_fn () const`

Return current const comb_probe_fn.

Definition at line 88 of file gp_ht_map.hpp.

5.267.3.4 `template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc , bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy > Eq_Fn & __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >::get_eq_fn ()`

Return current eq_fn.

Definition at line 58 of file gp_ht_map.hpp.

5.267.3.5 `template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc , bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy > const Eq_Fn & __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >::get_eq_fn () const`

Return current const eq_fn.

Definition at line 64 of file gp_ht_map.hpp.

5.267.3.6 `template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc , bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy > Hash_Fn & __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >::get_hash_fn ()`

Return current hash_fn.

Definition at line 46 of file gp_ht_map.hpp.

5.267.3.7 `template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc , bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy > const Hash_Fn & __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >::get_hash_fn () const`

Return current const hash_fn.

Definition at line 52 of file gp_ht_map.hpp.

5.267.3.8 `template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc , bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy > Probe_Fn & __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >::get_probe_fn ()`

Return current probe_fn.

Definition at line 70 of file gp_ht_map.hpp.

5.267.3.9 `template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc , bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy > const Probe_Fn & __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >::get_probe_fn () const`

Return current const probe_fn.

Definition at line 76 of file `gp_ht_map.hpp`.

5.267.3.10 `template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc , bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy > Resize_Policy & __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >::get_resize_policy ()`

Return current resize_policy.

Definition at line 94 of file `gp_ht_map.hpp`.

5.267.3.11 `template<typename Key , typename Mapped , typename Hash_Fn , typename Eq_Fn , typename _Alloc , bool Store_Hash, typename Comb_Probe_Fn , typename Probe_Fn , typename Resize_Policy > const Resize_Policy & __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >::get_resize_policy () const`

Return current const resize_policy.

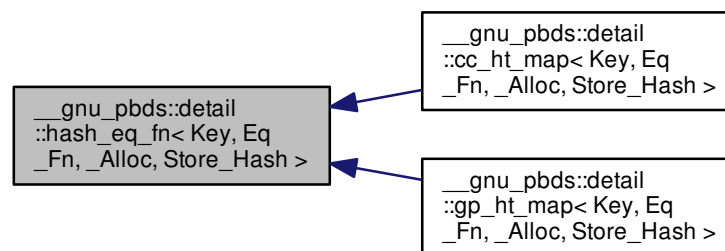
Definition at line 100 of file `gp_ht_map.hpp`.

The documentation for this class was generated from the following file:

- [gp_ht_map.hpp](#)

5.268 `__gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, Store_Hash >` Struct Template Reference

Inheritance diagram for `__gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, Store_Hash >`:



5.268.1 Detailed Description

```
template<typename Key, typename Eq_Fn, typename _Alloc, bool Store_Hash>
struct __gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, Store_Hash >
```

Primary template.

Definition at line 54 of file hash_eq_fn.hpp.

The documentation for this struct was generated from the following file:

- [hash_eq_fn.hpp](#)

5.269 __gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, false > Struct Template Reference

Inherits Eq_Fn.

Public Types

- typedef Eq_Fn **eq_fn_base**
- typedef _Alloc::template rebind< Key >::other **key_allocator**
- typedef key_allocator::const_reference **key_const_reference**

Public Member Functions

- **hash_eq_fn** (const Eq_Fn &r_eq_fn)
- bool **operator()** (key_const_reference r_lhs_key, key_const_reference r_rhs_key) const
- void **swap** (const [hash_eq_fn](#) &other)

5.269.1 Detailed Description

```
template<typename Key, typename Eq_Fn, typename _Alloc>
struct __gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, false >
```

Specialization 1 - The client requests that hash values not be stored.

Definition at line 58 of file hash_eq_fn.hpp.

The documentation for this struct was generated from the following file:

- [hash_eq_fn.hpp](#)

5.270 `__gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, true >` Struct Template Reference

Inherits `Eq_Fn`.

Public Types

- typedef `Eq_Fn` **eq_fn_base**
- typedef `_Alloc::template rebind< Key >::other` **key_allocator**
- typedef `key_allocator::const_reference` **key_const_reference**
- typedef `_Alloc::size_type` **size_type**

Public Member Functions

- **hash_eq_fn** (`const Eq_Fn &r_eq_fn`)
- bool **operator()** (`key_const_reference r_lhs_key, size_type lhs_hash, key_const_reference r_rhs_key, size_type rhs_hash`) const
- void **swap** (`const hash_eq_fn &other`)

5.270.1 Detailed Description

```
template<typename Key, class Eq_Fn, class _Alloc>
struct __gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc, true >
```

Specialization 2 - The client requests that hash values be stored.

Definition at line 81 of file `hash_eq_fn.hpp`.

The documentation for this struct was generated from the following file:

- [hash_eq_fn.hpp](#)

5.271 `__gnu_pbds::detail::hash_load_check_resize_trigger_size_base< Size_Type, Hold_Size >` Class Template Reference

5.271.1 Detailed Description

```
template<typename Size_Type, bool Hold_Size>
class __gnu_pbds::detail::hash_load_check_resize_trigger_size_base< Size_Type, Hold_Size >
```

Primary template.

Definition at line 50 of file `hash_load_check_resize_trigger_size_base.hpp`.

The documentation for this class was generated from the following file:

- [hash_load_check_resize_trigger_size_base.hpp](#)

5.272 `__gnu_pbds::detail::hash_load_check_resize_trigger_size_base< Size_Type, true >` Class Template Reference

Protected Types

- typedef `Size_Type` **size_type**

Protected Member Functions

- `size_type` **get_size** () const
- void **set_size** (size_type size)
- void **swap** ([hash_load_check_resize_trigger_size_base](#) &other)

5.272.1 Detailed Description

```
template<typename Size_Type>
class __gnu_pbds::detail::hash_load_check_resize_trigger_size_base< Size_Type, true >
```

Specializations.

Definition at line 54 of file `hash_load_check_resize_trigger_size_base.hpp`.

The documentation for this class was generated from the following file:

- [hash_load_check_resize_trigger_size_base.hpp](#)

5.273 `__gnu_pbds::detail::left_child_next_sibling_heap< Value_Type, Cmp_Fn, Node_Metadata, _Alloc >` Class Template Reference

Inherits `Cmp_Fn`.

Public Types

- typedef `_Alloc` **allocator_type**
- typedef `Cmp_Fn` **cmp_fn**
- typedef [left_child_next_sibling_heap_const_iterator_< node, _Alloc >](#) **const_iterator**
- typedef `__rebind_v::other::const_pointer` **const_pointer**
- typedef `__rebind_v::other::const_reference` **const_reference**
- typedef `_Alloc::difference_type` **difference_type**
- typedef [const_iterator](#) **iterator**
- typedef [left_child_next_sibling_heap_node_point_const_iterator_< node, _Alloc >](#) **point_const_iterator**
- typedef [point_const_iterator](#) **point_iterator**
- typedef `__rebind_v::other::pointer` **pointer**
- typedef `__rebind_v::other::reference` **reference**
- typedef `_Alloc::size_type` **size_type**
- typedef `Value_Type` **value_type**

Public Member Functions

- `left_child_next_sibling_heap` (const Cmp_Fn &)
- `left_child_next_sibling_heap` (const `left_child_next_sibling_heap` &)
- `iterator begin` ()
- `const_iterator begin` () const
- void `clear` ()
- bool `empty` () const
- `iterator end` ()
- `const_iterator end` () const
- Cmp_Fn & `get_cmp_fn` ()
- const Cmp_Fn & `get_cmp_fn` () const
- size_type `max_size` () const
- size_type `size` () const
- void `swap` (`left_child_next_sibling_heap`< Value_Type, Cmp_Fn, Node_Metadata, _Alloc > &)

Protected Types

- typedef node_allocator::value_type `node`
- typedef _Alloc::template rebind< `left_child_next_sibling_heap_node`< Value_Type, Node_Metadata, _Alloc >::other `node_allocator`
- typedef node_allocator::const_pointer `node_const_pointer`
- typedef Node_Metadata `node_metadata`
- typedef node_allocator::pointer `node_pointer`
- typedef `std::pair`< node_pointer, node_pointer > `node_pointer_pair`

Protected Member Functions

- void `actual_erase_node` (node_pointer)
- void `bubble_to_top` (node_pointer)
- void `clear_imp` (node_pointer)
- node_pointer `get_new_node_for_insert` (const_reference)
- template<typename Pred >
node_pointer `prune` (Pred)
- void `swap_with_parent` (node_pointer, node_pointer)
- void `to_linked_list` ()
- void `value_swap` (`left_child_next_sibling_heap` &)

Static Protected Member Functions

- static void `make_child_of` (node_pointer, node_pointer)
- static node_pointer `parent` (node_pointer)

Protected Attributes

- node_pointer `m_p_root`
- size_type `m_size`

5.273.1 Detailed Description

```
template<typename Value_Type, typename Cmp_Fn, typename Node_Metadata, typename _Alloc>
class __gnu_pbds::detail::left_child_next_sibling_heap< Value_Type, Cmp_Fn, Node_Metadata, _Alloc >
```

Base class for a basic heap.

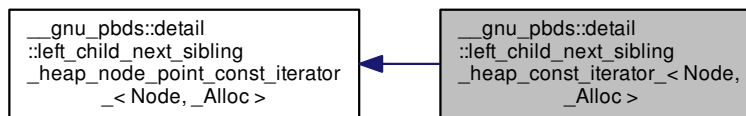
Definition at line 90 of file `left_child_next_sibling_heap_.hpp`.

The documentation for this class was generated from the following file:

- [left_child_next_sibling_heap_.hpp](#)

5.274 __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_< Node, _Alloc > Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_< Node, _Alloc >`:



Public Types

- typedef [base_type::const_pointer](#) `const_pointer`
- typedef [base_type::const_reference](#) `const_reference`
- typedef `_Alloc::difference_type` `difference_type`
- typedef [std::forward_iterator_tag](#) `iterator_category`
- typedef [base_type::pointer](#) `pointer`
- typedef [base_type::reference](#) `reference`
- typedef [base_type::value_type](#) `value_type`

Public Member Functions

- **left_child_next_sibling_heap_const_iterator_** (node_pointer p_nd)
- [left_child_next_sibling_heap_const_iterator_](#) ()
- [left_child_next_sibling_heap_const_iterator_](#) (const [left_child_next_sibling_heap_const_iterator_](#)< Node, _Alloc > &other)
- bool **operator!=** (const [left_child_next_sibling_heap_const_iterator_](#)< Node, _Alloc > &other) const
- bool **operator!=** (const [left_child_next_sibling_heap_node_point_const_iterator_](#)< Node, _Alloc > &other) const
- `const_reference operator*` () const
- [left_child_next_sibling_heap_const_iterator_](#)< Node, _Alloc > & **operator++** ()
- [left_child_next_sibling_heap_const_iterator_](#)< Node, _Alloc > **operator++** (int)
- `const_pointer operator->` () const
- bool **operator==** (const [left_child_next_sibling_heap_const_iterator_](#)< Node, _Alloc > &other) const
- bool **operator==** (const [left_child_next_sibling_heap_node_point_const_iterator_](#)< Node, _Alloc > &other) const

Public Attributes

- node_pointer `m_p_nd`

5.274.1 Detailed Description

```
template<typename Node, typename _Alloc>
class __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_< Node, _Alloc >
```

Const point-type iterator.

Definition at line 60 of file `left_child_next_sibling_heap_/const_iterator.hpp`.

5.274.2 Member Typedef Documentation

```
5.274.2.1 template<typename Node , typename _Alloc > typedef base_type::const_pointer
__gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_< Node, _Alloc >::const_pointer
```

Iterator's const pointer type.

Definition at line 81 of file `left_child_next_sibling_heap_/const_iterator.hpp`.

```
5.274.2.2 template<typename Node , typename _Alloc > typedef base_type::const_reference
__gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_< Node, _Alloc >::const_reference
```

Iterator's const reference type.

Definition at line 87 of file `left_child_next_sibling_heap_/const_iterator.hpp`.

```
5.274.2.3 template<typename Node , typename _Alloc > typedef _Alloc::difference_type __gnu_pbds::detail::left_child_↵
next_sibling_heap_const_iterator_< Node, _Alloc >::difference_type
```

Difference type.

Definition at line 72 of file `left_child_next_sibling_heap_/const_iterator.hpp`.

```
5.274.2.4 template<typename Node , typename _Alloc > typedef std::forward_iterator_tag
__gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_< Node, _Alloc >::iterator_category
```

Category.

Definition at line 69 of file `left_child_next_sibling_heap_/const_iterator.hpp`.

```
5.274.2.5 template<typename Node , typename _Alloc > typedef base_type::pointer __gnu_pbds::detail::left_child_↵
next_sibling_heap_const_iterator_< Node, _Alloc >::pointer
```

Iterator's pointer type.

Definition at line 78 of file `left_child_next_sibling_heap_/const_iterator.hpp`.

5.274.2.6 `template<typename Node , typename _Alloc > typedef base_type::reference __gnu_pbds::detail::left_child_↵
next_sibling_heap_const_iterator_< Node, _Alloc >::reference`

Iterator's reference type.

Definition at line 84 of file `left_child_next_sibling_heap_/const_iterator.hpp`.

5.274.2.7 `template<typename Node , typename _Alloc > typedef base_type::value_type
__gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_< Node, _Alloc >::value_type`

Iterator's value type.

Definition at line 75 of file `left_child_next_sibling_heap_/const_iterator.hpp`.

5.274.3 Constructor & Destructor Documentation

5.274.3.1 `template<typename Node , typename _Alloc > __gnu_pbds::detail::left_child_next_sibling_↵
heap_const_iterator_< Node, _Alloc >::left_child_next_sibling_heap_const_iterator_ ()
[inline]`

Default constructor.

Definition at line 96 of file `left_child_next_sibling_heap_/const_iterator.hpp`.

5.274.3.2 `template<typename Node , typename _Alloc > __gnu_pbds::detail::left_child_next_sibling_↵
heap_const_iterator_< Node, _Alloc >::left_child_next_sibling_heap_const_iterator_ (const
left_child_next_sibling_heap_const_iterator_< Node, _Alloc > & other) [inline]`

Copy constructor.

Definition at line 101 of file `left_child_next_sibling_heap_/const_iterator.hpp`.

5.274.4 Member Function Documentation

5.274.4.1 `template<typename Node , typename _Alloc > bool __gnu_pbds::detail::left_child_next_sibling_heap_↵
const_iterator_< Node, _Alloc >::operator!= (const left_child_next_sibling_heap_const_iterator_< Node,
_Alloc > & other) const [inline]`

Compares content (negatively) to a different iterator object.

Definition at line 111 of file `left_child_next_sibling_heap_/const_iterator.hpp`.

5.274.4.2 `template<typename Node , typename _Alloc > bool __gnu_pbds::detail::left_child_↵
next_sibling_heap_node_point_const_iterator_< Node, _Alloc >::operator!= (const
left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc > & other) const [inline],
[inherited]`

Compares content (negatively) to a different iterator object.

Definition at line 137 of file `left_child_next_sibling_heap_/point_const_iterator.hpp`.

5.274.4.3 `template<typename Node , typename _Alloc > const_reference __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc >::operator*() const` `[inline]`,
`[inherited]`

Access.

Definition at line 124 of file `left_child_next_sibling_heap_/point_const_iterator.hpp`.

5.274.4.4 `template<typename Node , typename _Alloc > const_pointer __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc >::operator-> () const` `[inline]`,
`[inherited]`

Access.

Definition at line 116 of file `left_child_next_sibling_heap_/point_const_iterator.hpp`.

5.274.4.5 `template<typename Node , typename _Alloc > bool __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_< Node, _Alloc >::operator==(const left_child_next_sibling_heap_const_iterator_< Node, _Alloc > & other) const` `[inline]`

Compares content to a different iterator object.

Definition at line 106 of file `left_child_next_sibling_heap_/const_iterator.hpp`.

5.274.4.6 `template<typename Node , typename _Alloc > bool __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc >::operator==(const left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc > & other) const` `[inline]`,
`[inherited]`

Compares content to a different iterator object.

Definition at line 132 of file `left_child_next_sibling_heap_/point_const_iterator.hpp`.

The documentation for this class was generated from the following file:

- [left_child_next_sibling_heap_/const_iterator.hpp](#)

5.275 `__gnu_pbds::detail::left_child_next_sibling_heap_node_<_Value,_Metadata,_Alloc>` Struct Template Reference

Public Types

- typedef `_Metadata` **metadata_type**
- typedef `_Alloc::template rebind< this_type >::other::pointer` **node_pointer**
- typedef `_Alloc::size_type` **size_type**
- typedef `_Value` **value_type**

Public Attributes

- metadata_type **m_metadata**
- node_pointer **m_p_l_child**
- node_pointer **m_p_next_sibling**
- node_pointer **m_p_prev_or_parent**
- value_type **m_value**

5.275.1 Detailed Description

```
template<typename _Value, typename _Metadata, typename _Alloc>
struct __gnu_pbds::detail::left_child_next_sibling_heap_node_< _Value, _Metadata, _Alloc >
```

Node.

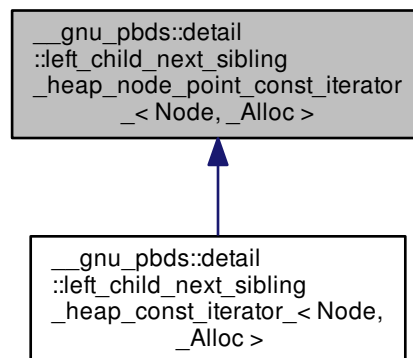
Definition at line 50 of file `left_child_next_sibling_heap_/node.hpp`.

The documentation for this struct was generated from the following file:

- [left_child_next_sibling_heap_/node.hpp](#)

5.276 __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc > Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc >`:



Public Types

- `typedef _Alloc::template rebind< value_type >::other::const_pointer const_pointer`
- `typedef _Alloc::template rebind< value_type >::other::const_reference const_reference`
- `typedef trivial_iterator_difference_type difference_type`
- `typedef trivial_iterator_tag iterator_category`
- `typedef _Alloc::template rebind< value_type >::other::pointer pointer`
- `typedef _Alloc::template rebind< value_type >::other::reference reference`
- `typedef Node::value_type value_type`

Public Member Functions

- `left_child_next_sibling_heap_node_point_const_iterator_ (node_pointer p_nd)`
- `left_child_next_sibling_heap_node_point_const_iterator_ ()`
- `left_child_next_sibling_heap_node_point_const_iterator_ (const left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc > &other)`
- `bool operator!= (const left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc > &other) const`
- `const_reference operator* () const`
- `const_pointer operator-> () const`
- `bool operator== (const left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc > &other) const`

Public Attributes

- node_pointer **`m_p_nd`**

Protected Types

- `typedef _Alloc::template rebind< Node >::other::pointer node_pointer`

5.276.1 Detailed Description

```
template<typename Node, typename _Alloc>
class __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc >
```

Const point-type iterator.

Definition at line 61 of file `left_child_next_sibling_heap_/point_const_iterator.hpp`.

5.276.2 Member Typedef Documentation

5.276.2.1 `template<typename Node , typename _Alloc > typedef _Alloc::template rebind< value_type >::other::const_pointer __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc >::const_pointer`

Iterator's const pointer type.

Definition at line 86 of file `left_child_next_sibling_heap_/point_const_iterator.hpp`.

```
5.276.2.2  template<typename Node , typename _Alloc > typedef _Alloc::template rebind< value_type>::other::const_reference
__gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc
>::const_reference
```

Iterator's const reference type.

Definition at line 98 of file left_child_next_sibling_heap_/point_const_iterator.hpp.

```
5.276.2.3  template<typename Node , typename _Alloc > typedef trivial_iterator_difference_type
__gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc
>::difference_type
```

Difference type.

Definition at line 71 of file left_child_next_sibling_heap_/point_const_iterator.hpp.

```
5.276.2.4  template<typename Node , typename _Alloc > typedef trivial_iterator_tag __gnu_pbds::
::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc
>::iterator_category
```

Category.

Definition at line 68 of file left_child_next_sibling_heap_/point_const_iterator.hpp.

```
5.276.2.5  template<typename Node , typename _Alloc > typedef _Alloc::template rebind< value_type>::other::pointer
__gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc
>::pointer
```

Iterator's pointer type.

Definition at line 80 of file left_child_next_sibling_heap_/point_const_iterator.hpp.

```
5.276.2.6  template<typename Node , typename _Alloc > typedef _Alloc::template rebind< value_type>::other::reference
__gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc
>::reference
```

Iterator's reference type.

Definition at line 92 of file left_child_next_sibling_heap_/point_const_iterator.hpp.

```
5.276.2.7  template<typename Node , typename _Alloc > typedef Node::value_type __gnu_pbds::
::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc
>::value_type
```

Iterator's value type.

Definition at line 74 of file left_child_next_sibling_heap_/point_const_iterator.hpp.

5.276.3 Constructor & Destructor Documentation

5.276.3.1 `template<typename Node , typename _Alloc > __gnu_pbds::detail::left_child_next_sibling_heap_node_↵
point_const_iterator_< Node, _Alloc >::left_child_next_sibling_heap_node_point_const_iterator_ ()
[inline]`

Default constructor.

Definition at line 106 of file `left_child_next_sibling_heap_/point_const_iterator.hpp`.

5.276.3.2 `template<typename Node , typename _Alloc > __gnu_pbds::detail::left_child_next_sibling_heap_node_↵
point_const_iterator_< Node, _Alloc >::left_child_next_sibling_heap_node_point_const_iterator_ (
const left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc > & other) [inline]`

Copy constructor.

Definition at line 111 of file `left_child_next_sibling_heap_/point_const_iterator.hpp`.

5.276.4 Member Function Documentation

5.276.4.1 `template<typename Node , typename _Alloc > bool __gnu_pbds::detail::left_child_↵
next_sibling_heap_node_point_const_iterator_< Node, _Alloc >::operator!= (const
left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc > & other) const [inline]`

Compares content (negatively) to a different iterator object.

Definition at line 137 of file `left_child_next_sibling_heap_/point_const_iterator.hpp`.

5.276.4.2 `template<typename Node , typename _Alloc > const_reference __gnu_pbds::detail::left_↵
child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc >::operator* () const
[inline]`

Access.

Definition at line 124 of file `left_child_next_sibling_heap_/point_const_iterator.hpp`.

5.276.4.3 `template<typename Node , typename _Alloc > const_pointer __gnu_pbds::detail::left_child_↵
_next_sibling_heap_node_point_const_iterator_< Node, _Alloc >::operator-> () const
[inline]`

Access.

Definition at line 116 of file `left_child_next_sibling_heap_/point_const_iterator.hpp`.

```
5.276.4.4  template<typename Node , typename _Alloc > bool __gnu_pbds::detail::left_child_↵
            next_sibling_heap_node_point_const_iterator_< Node, _Alloc >::operator==( const
            left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc > & other ) const  [inline]
```

Compares content to a different iterator object.

Definition at line 132 of file `left_child_next_sibling_heap_/point_const_iterator.hpp`.

The documentation for this class was generated from the following file:

- [left_child_next_sibling_heap_/point_const_iterator.hpp](#)

5.277 __gnu_pbds::detail::lu_counter_metadata< Size_Type > Class Template Reference

Public Types

- typedef Size_Type **size_type**

Friends

- class **lu_counter_policy_base**< **size_type** >

5.277.1 Detailed Description

```
template<typename Size_Type = std::size_t>
class __gnu_pbds::detail::lu_counter_metadata< Size_Type >
```

A list-update metadata type that moves elements to the front of the list based on the counter algorithm.

Definition at line 51 of file `lu_counter_metadata.hpp`.

The documentation for this class was generated from the following file:

- [lu_counter_metadata.hpp](#)

5.278 __gnu_pbds::detail::lu_counter_policy_base< Size_Type > Class Template Reference

Protected Types

- typedef Size_Type **size_type**

Protected Member Functions

- [lu_counter_metadata](#)< size_type > **operator()** (size_type max_size) const
- template<typename Metadata_Reference >
bool **operator()** (Metadata_Reference r_data, size_type m_max_count) const

5.279 `__gnu_pbds::detail::lu_map< Key, Mapped, Eq_Fn, _Alloc, Update_Policy >` Class Template Reference

5.278.1 Detailed Description

```
template<typename Size_Type>
class __gnu_pbds::detail::lu_counter_policy_base< Size_Type >
```

Base class for list-update counter policy.

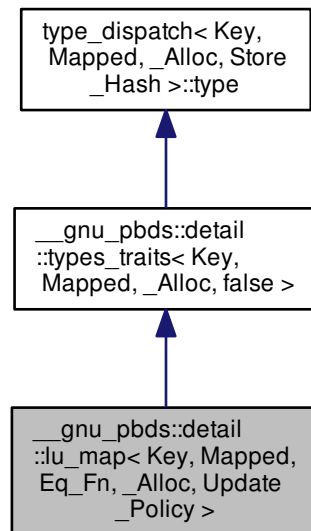
Definition at line 46 of file `lu_counter_metadata.hpp`.

The documentation for this class was generated from the following file:

- [lu_counter_metadata.hpp](#)

5.279 `__gnu_pbds::detail::lu_map< Key, Mapped, Eq_Fn, _Alloc, Update_Policy >` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::lu_map< Key, Mapped, Eq_Fn, _Alloc, Update_Policy >`:



Public Types

- typedef `_Alloc` **allocator_type**
- typedef `std::pair< size_type, size_type >` **comp_hash**
- typedef `const_iterator` **const_iterator**
- typedef `traits_base::const_pointer` **const_pointer**
- typedef `traits_base::const_reference` **const_reference**

- typedef _Alloc::difference_type **difference_type**
- typedef Eq_Fn **eq_fn**
- typedef iterator **iterator**
- typedef traits_base::key_const_pointer **key_const_pointer**
- typedef traits_base::key_const_reference **key_const_reference**
- typedef traits_base::key_pointer **key_pointer**
- typedef traits_base::key_reference **key_reference**
- typedef traits_base::key_type **key_type**
- typedef traits_base::mapped_const_pointer **mapped_const_pointer**
- typedef traits_base::mapped_const_reference **mapped_const_reference**
- typedef traits_base::mapped_pointer **mapped_pointer**
- typedef traits_base::mapped_reference **mapped_reference**
- typedef traits_base::mapped_type **mapped_type**
- typedef __nothrowcopy::indicator **no_throw_indicator**
- typedef point_const_iterator **point_const_iterator**
- typedef point_iterator **point_iterator**
- typedef traits_base::pointer **pointer**
- typedef traits_base::reference **reference**
- typedef _Alloc::size_type **size_type**
- typedef integral_constant< int, Store_Hash > **store_extra**
- typedef Update_Policy::metadata_type **update_metadata**
- typedef Update_Policy **update_policy**
- typedef traits_base::value_type **value_type**

Public Member Functions

- **lu_map** (const **lu_map**< Key, Mapped, Eq_Fn, _Alloc, Update_Policy > &)
- template<typename It >
 lu_map (It, It)
- iterator **begin** ()
- const_iterator **begin** () const
- void **clear** ()
- bool **empty** () const
- iterator **end** ()
- const_iterator **end** () const
- bool **erase** (key_const_reference)
- template<typename Pred >
 size_type **erase_if** (Pred)
- point_iterator **find** (key_const_reference r_key)
- point_const_iterator **find** (key_const_reference r_key) const
- **std::pair**< point_iterator, bool > **insert** (const_reference)
- size_type **max_size** () const
- mapped_reference **operator[]** (key_const_reference r_key)
- size_type **size** () const
- void **swap** (**lu_map**< Key, Mapped, Eq_Fn, _Alloc, Update_Policy > &)

Public Attributes

- no_throw_indicator **m_no_throw_copies_indicator**
- store_extra **m_store_extra_indicator**

Protected Member Functions

- `template<typename It >`
`void copy_from_range (It, It)`

Friends

- class `const_iterator_`
- class `iterator_`

5.279.1 Detailed Description

```
template<typename Key, typename Mapped, typename Eq_Fn, typename _Alloc, typename Update_Policy>
class __gnu_pbds::detail::lu_map< Key, Mapped, Eq_Fn, _Alloc, Update_Policy >
```

list-based (with updates) associative container. Skip to the lu, my darling.

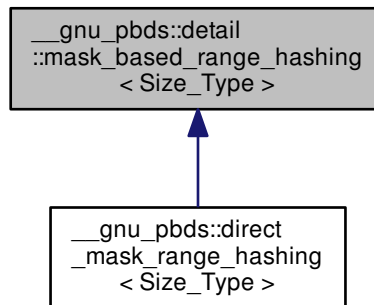
Definition at line 91 of file `lu_map.hpp`.

The documentation for this class was generated from the following file:

- [lu_map.hpp](#)

5.280 `__gnu_pbds::detail::mask_based_range_hashing< Size_Type >` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::mask_based_range_hashing< Size_Type >`:



Protected Types

- `typedef Size_Type` **size_type**

Protected Member Functions

- void **notify_resized** (size_type size)
- size_type **range_hash** (size_type hash) const
- void **swap** ([mask_based_range_hashing](#) &other)

5.280.1 Detailed Description

```
template<typename Size_Type>
class __gnu_pbds::detail::mask_based_range_hashing< Size_Type >
```

Range hashing policy.

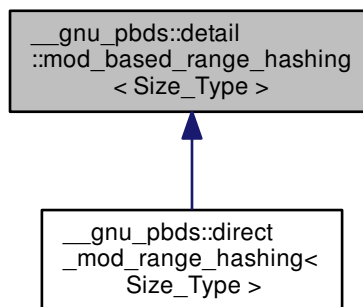
Definition at line 50 of file mask_based_range_hashing.hpp.

The documentation for this class was generated from the following file:

- [mask_based_range_hashing.hpp](#)

5.281 __gnu_pbds::detail::mod_based_range_hashing< Size_Type > Class Template Reference

Inheritance diagram for __gnu_pbds::detail::mod_based_range_hashing< Size_Type >:



Protected Types

- typedef Size_Type **size_type**

Protected Member Functions

- void **notify_resized** (size_type s)
- size_type **range_hash** (size_type s) const
- void **swap** ([mod_based_range_hashing](#) &other)

5.281.1 Detailed Description

```
template<typename Size_Type>  
class __gnu_pbds::detail::mod_based_range_hashing< Size_Type >
```

Mod based range hashing.

Definition at line 50 of file `mod_based_range_hashing.hpp`.

The documentation for this class was generated from the following file:

- [mod_based_range_hashing.hpp](#)

5.282 `__gnu_pbds::detail::no_throw_copies< Key, Mapped >` Struct Template Reference

Public Types

- typedef integral_constant< int, __simple > **indicator**

Static Public Attributes

- static const bool **__simple**

5.282.1 Detailed Description

```
template<typename Key, typename Mapped>  
struct __gnu_pbds::detail::no_throw_copies< Key, Mapped >
```

Primary template.

Definition at line 61 of file `types_traits.hpp`.

The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

5.283 `__gnu_pbds::detail::no_throw_copies< Key, null_type >` Struct Template Reference

Public Types

- `typedef integral_constant< int, is_simple< Key >::value > indicator`

5.283.1 Detailed Description

```
template<typename Key>
struct __gnu_pbds::detail::no_throw_copies< Key, null_type >
```

Specialization.

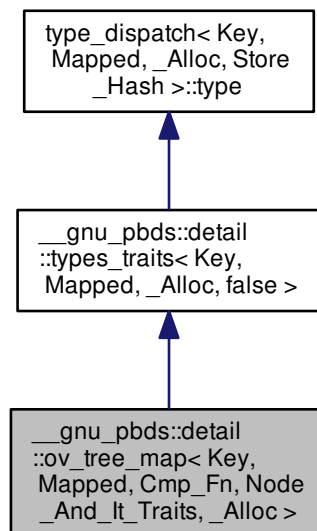
Definition at line 70 of file `types_traits.hpp`.

The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

5.284 `__gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >`:



Classes

- class [cond_dtor](#)

Public Types

- typedef `_Alloc` **allocator_type**
- typedef `Cmp_Fn` **cmp_fn**
- typedef `std::pair< size_type, size_type >` **comp_hash**
- typedef `point_const_iterator` **const_iterator**
- typedef `traits_base::const_pointer` **const_pointer**
- typedef `traits_base::const_reference` **const_reference**
- typedef `ov_tree_tag` **container_category**
- typedef `_Alloc::difference_type` **difference_type**
- typedef `point_iterator` **iterator**
- typedef `traits_base::key_const_pointer` **key_const_pointer**
- typedef `traits_base::key_const_reference` **key_const_reference**
- typedef `traits_base::key_pointer` **key_pointer**
- typedef `traits_base::key_reference` **key_reference**
- typedef `traits_base::key_type` **key_type**
- typedef `traits_base::mapped_const_pointer` **mapped_const_pointer**
- typedef `traits_base::mapped_const_reference` **mapped_const_reference**
- typedef `traits_base::mapped_pointer` **mapped_pointer**
- typedef `traits_base::mapped_reference` **mapped_reference**
- typedef `traits_base::mapped_type` **mapped_type**
- typedef `__nothrowcopy::indicator` **no_throw_indicator**
- typedef `traits_type::node_const_iterator` **node_const_iterator**
- typedef `traits_type::node_iterator` **node_iterator**
- typedef `traits_type::node_update` **node_update**
- typedef `const_pointer` **point_const_iterator**
- typedef `pointer` **point_iterator**
- typedef `traits_base::pointer` **pointer**
- typedef `traits_base::reference` **reference**
- typedef `_Alloc::size_type` **size_type**
- typedef `integral_constant< int, Store_Hash >` **store_extra**
- typedef `traits_base::value_type` **value_type**

Public Member Functions

- **ov_tree_map** (`const Cmp_Fn &`)
- **ov_tree_map** (`const Cmp_Fn &, const node_update &`)
- **ov_tree_map** (`const ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &`)
- iterator **begin** ()
- `const_iterator` **begin** () const
- void **clear** ()
- template<typename It >
void **copy_from_range** (It, It)
- bool **empty** () const
- iterator **end** ()

- `const_iterator` **end** () const
- `bool` **erase** (key_const_reference)
- `iterator` **erase** (iterator it)
- `template<typename Pred >`
`size_type` **erase_if** (Pred)
- `point_iterator` **find** (key_const_reference r_key)
- `point_const_iterator` **find** (key_const_reference r_key) const
- `Cmp_Fn` & **get_cmp_fn** ()
- `const Cmp_Fn` & **get_cmp_fn** () const
- `std::pair< point_iterator, bool >` **insert** (const_reference r_value)
- `void` **join** (`ov_tree_map`< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)
- `point_iterator` **lower_bound** (key_const_reference r_key)
- `point_const_iterator` **lower_bound** (key_const_reference r_key) const
- `size_type` **max_size** () const
- `node_const_iterator` **node_begin** () const
- `node_iterator` **node_begin** ()
- `node_const_iterator` **node_end** () const
- `node_iterator` **node_end** ()
- `mapped_reference` **operator[]** (key_const_reference r_key)
- `size_type` **size** () const
- `void` **split** (key_const_reference, `ov_tree_map`< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)
- `void` **swap** (`ov_tree_map`< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)
- `point_iterator` **upper_bound** (key_const_reference r_key)
- `point_const_iterator` **upper_bound** (key_const_reference r_key) const

Public Attributes

- `no_throw_indicator` **m_no_throw_copies_indicator**
- `store_extra` **m_store_extra_indicator**

5.284.1 Detailed Description

```
template<typename Key, typename Mapped, typename Cmp_Fn, typename Node_And_It_Traits, typename _Alloc>
class __gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >
```

Ordered-vector tree associative-container.

Definition at line 106 of file `ov_tree_map.hpp`.

5.284.2 Member Function Documentation

5.284.2.1 `template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename _Alloc >` `ov_tree_map`< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::`node_const_iterator`
`__gnu_pbds::detail::ov_tree_map`< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::`node_begin` () const
`[inline]`

Returns a `const node_iterator` corresponding to the node at the root of the tree.

Definition at line 45 of file `ov_tree_map.hpp`.

5.284.2.2 `template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits ,
typename _Alloc > ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_iterator
__gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_begin ()
[inline]`

Returns a `node_iterator` corresponding to the node at the root of the tree.

Definition at line 57 of file `ov_tree_map.hpp`.

5.284.2.3 `template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename
_Alloc > ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_const_iterator
__gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_end () const
[inline]`

Returns a `const node_iterator` corresponding to a node just after a leaf of the tree.

Definition at line 51 of file `ov_tree_map.hpp`.

5.284.2.4 `template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits ,
typename _Alloc > ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_iterator
__gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_end ()
[inline]`

Returns a `node_iterator` corresponding to a node just after a leaf of the tree.

Definition at line 63 of file `ov_tree_map.hpp`.

The documentation for this class was generated from the following file:

- [ov_tree_map.hpp](#)

5.285 `__gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::cond_dtor< Size_Type >` Class Template Reference

Public Member Functions

- **cond_dtor** (value_vector `a_vec`, iterator `&r_last_it`, `Size_Type` `total_size`)
- void **set_no_action** ()

Protected Attributes

- value_vector **m_a_vec**
- const `Size_Type` **m_max_size**
- bool **m_no_action**
- iterator & **m_r_last_it**

5.285.1 Detailed Description

```
template<typename Key, typename Mapped, typename Cmp_Fn, typename Node_And_It_Traits, typename _Alloc>
template<typename Size_Type>
class __gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::cond_dtor< Size_Type >
```

Conditional destructor.

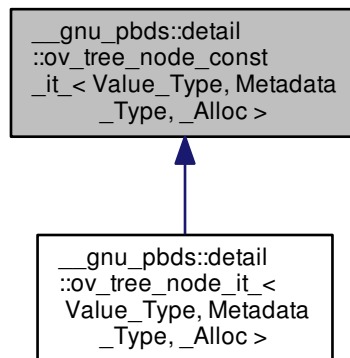
Definition at line 182 of file `ov_tree_map.hpp`.

The documentation for this class was generated from the following file:

- [ov_tree_map.hpp](#)

5.286 __gnu_pbds::detail::ov_tree_node_const_it< Value_Type, Metadata_Type, _Alloc > Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::ov_tree_node_const_it< Value_Type, Metadata_Type, _Alloc >`:



Public Types

- typedef `_Alloc::template rebind< typename remove_const< Value_Type >::type >::other::const_pointer` **const_reference**
- typedef `trivial_iterator_difference_type` **difference_type**
- typedef `trivial_iterator_tag` **iterator_category**
- typedef `_Alloc::template rebind< metadata_type >::other::const_reference` **metadata_const_reference**
- typedef `Metadata_Type` **metadata_type**
- typedef `_Alloc::template rebind< typename remove_const< Value_Type >::type >::other::const_pointer` **reference**
- typedef `_Alloc::template rebind< Value_Type >::other::const_pointer` **value_type**

Public Member Functions

- `ov_tree_node_const_it_` (const_pointer p_nd=0, const_pointer p_begin_nd=0, const_pointer p_end_nd=0, const_metadata_pointer p_metadata=0)
- `this_type get_l_child ()` const
- metadata_const_reference `get_metadata ()` const
- `this_type get_r_child ()` const
- bool `operator!=` (const `this_type` &other) const
- const_reference `operator*` () const
- bool `operator==` (const `this_type` &other) const

Public Attributes

- pointer `m_p_begin_value`
- pointer `m_p_end_value`
- const_metadata_pointer `m_p_metadata`
- pointer `m_p_value`

Protected Types

- typedef `_Alloc::template rebind< Metadata_Type >::other::const_pointer` `const_metadata_pointer`
- typedef `_Alloc::template rebind< Value_Type >::other::const_pointer` `const_pointer`
- typedef `_Alloc::template rebind< Value_Type >::other::pointer` `pointer`
- typedef `ov_tree_node_const_it_< Value_Type, Metadata_Type, _Alloc >` `this_type`

Static Protected Member Functions

- template<typename Ptr >
static Ptr `mid_pointer` (Ptr p_begin, Ptr p_end)

5.286.1 Detailed Description

```
template<typename Value_Type, typename Metadata_Type, typename _Alloc>
class __gnu_pbds::detail::ov_tree_node_const_it_< Value_Type, Metadata_Type, _Alloc >
```

Const node reference.

Definition at line 57 of file `ov_tree_map_/node_iterators.hpp`.

5.286.2 Member Function Documentation

5.286.2.1 `template<typename Value_Type , typename Metadata_Type , typename _Alloc > this_type`
`__gnu_pbds::detail::ov_tree_node_const_it_< Value_Type, Metadata_Type, _Alloc >::get_l_child ()` const
`[inline]`

Returns the node iterator associated with the left node.

Definition at line 142 of file `ov_tree_map_/node_iterators.hpp`.

5.286.2.2 `template<typename Value_Type , typename Metadata_Type , typename _Alloc > this_type
__gnu_pbds::detail::ov_tree_node_const_it_< Value_Type, Metadata_Type, _Alloc >::get_r_child () const
[inline]`

Returns the node iterator associated with the right node.

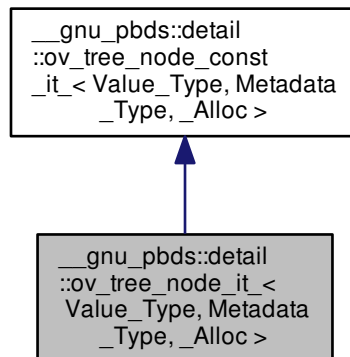
Definition at line 158 of file `ov_tree_map_/node_iterators.hpp`.

The documentation for this class was generated from the following file:

- [ov_tree_map_/node_iterators.hpp](#)

5.287 `__gnu_pbds::detail::ov_tree_node_it_< Value_Type, Metadata_Type, _Alloc >` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::ov_tree_node_it_< Value_Type, Metadata_Type, _Alloc >`:



Public Types

- `typedef _Alloc::template rebind< typename remove_const< Value_Type >::type >::other::pointer const_` ← **reference**
- `typedef trivial_iterator_difference_type difference_type`
- `typedef trivial_iterator_tag iterator_category`
- `typedef _Alloc::template rebind< metadata_type >::other::const_reference metadata_const_reference`
- `typedef Metadata_Type metadata_type`
- `typedef _Alloc::template rebind< typename remove_const< Value_Type >::type >::other::pointer reference`
- `typedef _Alloc::template rebind< Value_Type >::other::pointer value_type`

Public Member Functions

- `ov_tree_node_it_` (const_pointer p_nd=0, const_pointer p_begin_nd=0, const_pointer p_end_nd=0, const_pointer metadata_pointer p_metadata=0)
- `ov_tree_node_it_get_l_child` () const
- metadata_const_reference `get_metadata` () const
- `ov_tree_node_it_get_r_child` () const
- bool `operator!=` (const `this_type` &other) const
- reference `operator*` () const
- bool `operator==` (const `this_type` &other) const

Public Attributes

- pointer `m_p_begin_value`
- pointer `m_p_end_value`
- const_metadata_pointer `m_p_metadata`
- pointer `m_p_value`

Static Protected Member Functions

- template<typename Ptr >
static Ptr `mid_pointer` (Ptr p_begin, Ptr p_end)

5.287.1 Detailed Description

```
template<typename Value_Type, typename Metadata_Type, typename _Alloc>
class __gnu_pbds::detail::ov_tree_node_it_< Value_Type, Metadata_Type, _Alloc >
```

Node reference.

Definition at line 204 of file `ov_tree_map_/node_iterators.hpp`.

5.287.2 Member Function Documentation

5.287.2.1 `template<typename Value_Type , typename Metadata_Type , typename _Alloc > ov_tree_node_it_`
`__gnu_pbds::detail::ov_tree_node_it_< Value_Type, Metadata_Type, _Alloc >::get_l_child ()` const
`[inline]`

Returns the node reference associated with the left node.

Definition at line 252 of file `ov_tree_map_/node_iterators.hpp`.

5.287.2.2 `template<typename Value_Type , typename Metadata_Type , typename _Alloc > ov_tree_node_it_`
`__gnu_pbds::detail::ov_tree_node_it_< Value_Type, Metadata_Type, _Alloc >::get_r_child ()` const
`[inline]`

Returns the node reference associated with the right node.

Definition at line 268 of file `ov_tree_map_/node_iterators.hpp`.

5.287.2.3 `template<typename Value_Type , typename Metadata_Type , typename _Alloc > reference
 __gnu_pbds::detail::ov_tree_node_it_< Value_Type, Metadata_Type, _Alloc >::operator* () const
 [inline]`

Access.

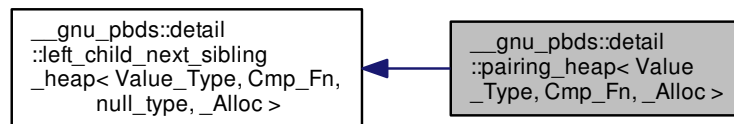
Definition at line 247 of file `ov_tree_map_/node_iterators.hpp`.

The documentation for this class was generated from the following file:

- [ov_tree_map_/node_iterators.hpp](#)

5.288 __gnu_pbds::detail::pairing_heap< Value_Type, Cmp_Fn, _Alloc > Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::pairing_heap< Value_Type, Cmp_Fn, _Alloc >`:



Public Types

- typedef `_Alloc` **allocator_type**
- typedef `Cmp_Fn` **cmp_fn**
- typedef `base_type::const_iterator` **const_iterator**
- typedef `__rebind_a::const_pointer` **const_pointer**
- typedef `__rebind_a::const_reference` **const_reference**
- typedef `_Alloc::difference_type` **difference_type**
- typedef `base_type::iterator` **iterator**
- typedef `base_type::point_const_iterator` **point_const_iterator**
- typedef `base_type::point_iterator` **point_iterator**
- typedef `__rebind_a::pointer` **pointer**
- typedef `__rebind_a::reference` **reference**
- typedef `_Alloc::size_type` **size_type**
- typedef `Value_Type` **value_type**

Public Member Functions

- **pairing_heap** (const Cmp_Fn &)
- **pairing_heap** (const [pairing_heap](#) &)
- **iterator begin** ()
- **const_iterator begin** () const
- void **clear** ()
- bool **empty** () const
- **iterator end** ()
- **const_iterator end** () const
- void **erase** ([point_iterator](#))
- template<typename Pred >
size_type **erase_if** (Pred)
- Cmp_Fn & **get_cmp_fn** ()
- const Cmp_Fn & **get_cmp_fn** () const
- void **join** ([pairing_heap](#) &)
- size_type **max_size** () const
- void **modify** ([point_iterator](#), const_reference)
- void **pop** ()
- [point_iterator](#) **push** (const_reference)
- size_type **size** () const
- template<typename Pred >
void **split** (Pred, [pairing_heap](#) &)
- void **swap** ([pairing_heap](#) &)
- void **swap** ([left_child_next_sibling_heap](#)< Value_Type, Cmp_Fn, [null_type](#), _Alloc > &)
- const_reference **top** () const

Protected Types

- typedef node_allocator::value_type **node**
- typedef _Alloc::template rebind< [left_child_next_sibling_heap_node](#)< Value_Type, [null_type](#), _Alloc > >::other
node_allocator
- typedef node_allocator::const_pointer **node_const_pointer**
- typedef [null_type](#) **node_metadata**
- typedef [std::pair](#)< node_pointer, node_pointer > **node_pointer_pair**

Protected Member Functions

- void **actual_erase_node** (node_pointer)
- void **bubble_to_top** (node_pointer)
- void **clear_imp** (node_pointer)
- template<typename It >
void **copy_from_range** (It, It)
- node_pointer **get_new_node_for_insert** (const_reference)
- node_pointer **prune** (Pred)
- void **swap_with_parent** (node_pointer, node_pointer)
- void **to_linked_list** ()
- void **value_swap** ([left_child_next_sibling_heap](#) &)

Static Protected Member Functions

- static void **make_child_of** (node_pointer, node_pointer)
- static node_pointer **parent** (node_pointer)

Protected Attributes

- node_pointer **m_p_root**
- size_type **m_size**

5.288.1 Detailed Description

```
template<typename Value_Type, typename Cmp_Fn, typename _Alloc>
class __gnu_pbds::detail::pairing_heap< Value_Type, Cmp_Fn, _Alloc >
```

Pairing heap.

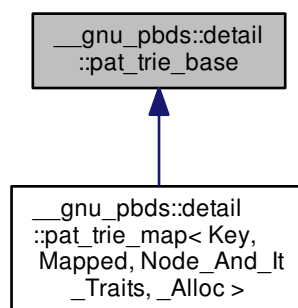
Definition at line 77 of file pairing_heap.hpp.

The documentation for this class was generated from the following file:

- [pairing_heap.hpp](#)

5.289 __gnu_pbds::detail::pat_trie_base Struct Reference

Inheritance diagram for __gnu_pbds::detail::pat_trie_base:



Classes

- class [_Clter](#)
- struct [_Head](#)
- struct [_Inode](#)
- class [_Iter](#)
- struct [_Leaf](#)
- struct [_Metadata](#)
- struct [_Metadata< null_type, _Alloc >](#)
- struct [_Node_base](#)
- class [_Node_citer](#)
- class [_Node_iter](#)

Public Types

- enum [node_type](#) { [i_node](#), [leaf_node](#), [head_node](#) }

5.289.1 Detailed Description

Base type for PATRICIA trees.

Definition at line 51 of file `pat_trie_base.hpp`.

5.289.2 Member Enumeration Documentation

5.289.2.1 enum `__gnu_pbds::detail::pat_trie_base::node_type`

Three types of nodes.

`i_node` is used by `_Inode`, `leaf_node` by `_Leaf`, and `head_node` by `_Head`.

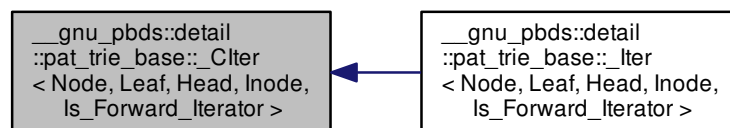
Definition at line 58 of file `pat_trie_base.hpp`.

The documentation for this struct was generated from the following file:

- [pat_trie_base.hpp](#)

5.290 `__gnu_pbds::detail::pat_trie_base::_Clter< Node, Leaf, Head, Inode, Is_Forward_Iterator > Class` Template Reference

Inheritance diagram for `__gnu_pbds::detail::pat_trie_base::_Clter< Node, Leaf, Head, Inode, Is_Forward_Iterator >`:



Public Types

- typedef __Alloc::template rebind< Head > **__rebind_h**
- typedef __Alloc::template rebind< Inode > **__rebind_in**
- typedef __Alloc::template rebind< Leaf > **__rebind_l**
- typedef __Alloc::template rebind< Node > **__rebind_n**
- typedef allocator_type **_Alloc**
- typedef Node::allocator_type **allocator_type**
- typedef type_traits::const_pointer **const_pointer**
- typedef type_traits::const_reference **const_reference**
- typedef allocator_type::difference_type **difference_type**
- typedef __rebind_h::other::pointer **head_pointer**
- typedef Inode::iterator **inode_iterator**
- typedef __rebind_in::other::pointer **inode_pointer**
- typedef [std::bidirectional_iterator_tag](#) **iterator_category**
- typedef __rebind_l::other::const_pointer **leaf_const_pointer**
- typedef __rebind_l::other::pointer **leaf_pointer**
- typedef __rebind_n::other::pointer **node_pointer**
- typedef type_traits::pointer **pointer**
- typedef type_traits::reference **reference**
- typedef Node::type_traits **type_traits**
- typedef type_traits::value_type **value_type**

Public Member Functions

- **_Clter** (node_pointer p_nd=0)
- **_Clter** (const [_Clter](#)< Node, Leaf, Head, Inode,!Is_Forward_Iterator > &other)
- bool **operator!=** (const [_Clter](#) &other) const
- bool **operator!=** (const [_Clter](#)< Node, Leaf, Head, Inode,!Is_Forward_Iterator > &other) const
- const_reference **operator*** () const
- [_Clter](#) & **operator++** ()
- [_Clter](#) **operator++** (int)
- [_Clter](#) & **operator--** ()
- [_Clter](#) **operator--** (int)
- const_pointer **operator->** () const
- [_Clter](#) & **operator=** (const [_Clter](#) &other)
- [_Clter](#) & **operator=** (const [_Clter](#)< Node, Leaf, Head, Inode,!Is_Forward_Iterator > &other)
- bool **operator==** (const [_Clter](#) &other) const
- bool **operator==** (const [_Clter](#)< Node, Leaf, Head, Inode,!Is_Forward_Iterator > &other) const

Public Attributes

- node_pointer **m_p_nd**

Protected Member Functions

- void **dec** (false_type)
- void **dec** (true_type)
- void **inc** (false_type)
- void **inc** (true_type)

Static Protected Member Functions

- static node_pointer **get_larger_sibling** (node_pointer p_nd)
- static node_pointer **get_smaller_sibling** (node_pointer p_nd)
- static leaf_pointer **leftmost_descendant** (node_pointer p_nd)
- static leaf_pointer **rightmost_descendant** (node_pointer p_nd)

5.290.1 Detailed Description

```
template<typename Node, typename Leaf, typename Head, typename Inode, bool Is_Forward_Iterator>
class __gnu_pbds::detail::pat_trie_base::_CIter< Node, Leaf, Head, Inode, Is_Forward_Iterator >
```

Const iterator.

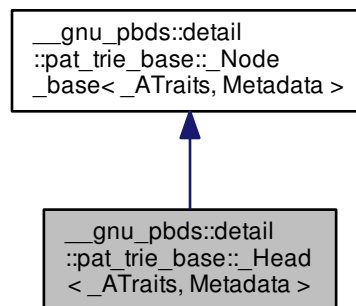
Definition at line 487 of file `pat_trie_base.hpp`.

The documentation for this class was generated from the following file:

- [pat_trie_base.hpp](#)

5.291 `__gnu_pbds::detail::pat_trie_base::_Head<_ATraits, Metadata>` Struct Template Reference

Inheritance diagram for `__gnu_pbds::detail::pat_trie_base::_Head<_ATraits, Metadata>`:



Public Types

- typedef `_Alloc::template rebind<_ATraits>` **__rebind_at**
- typedef `_Alloc::template rebind<_Node_base>` **__rebind_n**
- typedef `_ATraits::const_iterator` **a_const_iterator**
- typedef `_rebind_at::other::const_pointer` **a_const_pointer**
- typedef `_ATraits` **access_traits**
- typedef `_Alloc` **allocator_type**
- typedef `_Node_base<_ATraits, Metadata>` **base_type**
- typedef `base_type::node_pointer` **node_pointer**
- typedef `base_type::type_traits` **type_traits**

Public Attributes

- node_pointer **m_p_max**
- node_pointer **m_p_min**
- node_pointer **m_p_parent**
- const [node_type](#) **m_type**

5.291.1 Detailed Description

```
template<typename ATraits, typename Metadata>
struct __gnu_pbds::detail::pat_trie_base::_Head< ATraits, Metadata >
```

Head node for PATRICIA tree.

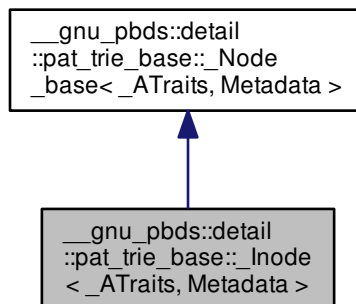
Definition at line 131 of file pat_trie_base.hpp.

The documentation for this struct was generated from the following file:

- [pat_trie_base.hpp](#)

5.292 __gnu_pbds::detail::pat_trie_base::_Inode< ATraits, Metadata > Struct Template Reference

Inheritance diagram for __gnu_pbds::detail::pat_trie_base::_Inode< ATraits, Metadata >:



Classes

- struct [const_iterator](#)
- struct [iterator](#)

Public Types

- enum { **arr_size** }
- typedef `_Alloc::template rebind<_ATraits>` **__rebind_at**
- typedef `_Alloc::template rebind<node_pointer>::other` **__rebind_np**
- typedef `base_type::allocator_type` **_Alloc**
- typedef `base_type::access_traits` **access_traits**
- typedef `_Alloc` **allocator_type**
- typedef `_Node_base<_ATraits, Metadata>` **base_type**
- typedef `__rebind_np::pointer` **node_pointer_pointer**
- typedef `__rebind_np::reference` **node_pointer_reference**
- typedef `_Alloc::size_type` **size_type**
- typedef `base_type::type_traits` **type_traits**
- typedef `type_traits::value_type` **value_type**

Public Member Functions

- **_Inode** (size_type, const a_const_iterator)
- node_pointer **add_child** (node_pointer, a_const_iterator, a_const_iterator, a_const_pointer)
- **const_iterator begin** () const
- **iterator begin** ()
- **const_iterator end** () const
- **iterator end** ()
- **iterator get_child_it** (a_const_iterator, a_const_iterator, a_const_pointer)
- node_pointer **get_child_node** (a_const_iterator, a_const_iterator, a_const_pointer)
- node_const_pointer **get_child_node** (a_const_iterator, a_const_iterator, a_const_pointer) const
- size_type **get_e_ind** () const
- node_const_pointer **get_join_child** (node_const_pointer, a_const_pointer) const
- node_pointer **get_join_child** (node_pointer, a_const_pointer)
- node_pointer **get_lower_bound_child_node** (a_const_iterator, a_const_iterator, size_type, a_const_pointer)
- leaf_pointer **leftmost_descendant** ()
- leaf_const_pointer **leftmost_descendant** () const
- a_const_iterator **pref_b_it** () const
- a_const_iterator **pref_e_it** () const
- void **remove_child** (node_pointer)
- void **remove_child** (iterator)
- void **replace_child** (node_pointer, a_const_iterator, a_const_iterator, a_const_pointer)
- leaf_pointer **rightmost_descendant** ()
- leaf_const_pointer **rightmost_descendant** () const
- bool **should_be_mine** (a_const_iterator, a_const_iterator, size_type, a_const_pointer) const
- void **update_prefixes** (a_const_pointer)

Public Attributes

- node_pointer **m_p_parent**
- const **node_type m_type**

5.292.1 Detailed Description

```
template<typename _ATraits, typename Metadata>
struct __gnu_pbds::detail::pat_trie_base::_Inode<_ATraits, Metadata >
```

Internal node type, PATRICIA tree.

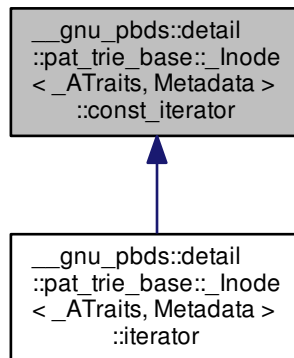
Definition at line 211 of file pat_trie_base.hpp.

The documentation for this struct was generated from the following file:

- [pat_trie_base.hpp](#)

5.293 __gnu_pbds::detail::pat_trie_base::_Inode<_ATraits, Metadata >::const_iterator Struct Reference

Inheritance diagram for __gnu_pbds::detail::pat_trie_base::_Inode<_ATraits, Metadata >::const_iterator:



Public Types

- typedef `_Alloc::difference_type` **difference_type**
- typedef `std::forward_iterator_tag` **iterator_category**
- typedef `node_pointer_pointer` **pointer**
- typedef `node_pointer_reference` **reference**
- typedef `node_pointer` **value_type**

Public Member Functions

- **const_iterator** (node_pointer_pointer p_p_cur=0, node_pointer_pointer p_p_end=0)
- bool **operator!=** (const [const_iterator](#) &other) const
- node_const_pointer **operator*** () const
- [const_iterator](#) & **operator++** ()
- [const_iterator](#) **operator++** (int)
- const node_pointer_pointer **operator->** () const
- bool **operator==** (const [const_iterator](#) &other) const

Public Attributes

- node_pointer_pointer **m_p_p_cur**
- node_pointer_pointer **m_p_p_end**

5.293.1 Detailed Description

```
template<typename _ATraits, typename Metadata>
struct __gnu_pbds::detail::pat_trie_base::_Inode<_ATraits, Metadata >::const_iterator
```

Constant child iterator.

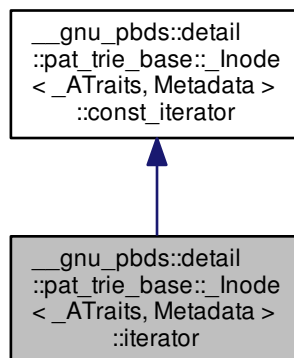
Definition at line 255 of file pat_trie_base.hpp.

The documentation for this struct was generated from the following file:

- [pat_trie_base.hpp](#)

5.294 __gnu_pbds::detail::pat_trie_base::_Inode<_ATraits, Metadata >::iterator Struct Reference

Inheritance diagram for __gnu_pbds::detail::pat_trie_base::_Inode<_ATraits, Metadata >::iterator:



Public Types

- typedef _Alloc::difference_type **difference_type**
- typedef [std::forward_iterator_tag](#) **iterator_category**
- typedef node_pointer_pointer **pointer**
- typedef node_pointer_reference **reference**
- typedef node_pointer **value_type**

Public Member Functions

- **iterator** (node_pointer_pointer p_p_cur=0, node_pointer_pointer p_p_end=0)
- bool **operator!=** (const [const_iterator](#) &other) const
- bool **operator!=** (const [iterator](#) &other) const
- node_const_pointer **operator*** () const
- node_pointer **operator*** ()
- [iterator](#) & **operator++** ()
- [iterator](#) **operator++** (int)
- const node_pointer_pointer **operator->** () const
- node_pointer_pointer **operator->** ()
- bool **operator==** (const [const_iterator](#) &other) const
- bool **operator==** (const [iterator](#) &other) const

Public Attributes

- node_pointer_pointer **m_p_p_cur**
- node_pointer_pointer **m_p_p_end**

5.294.1 Detailed Description

```
template<typename ATraits, typename Metadata>
struct __gnu_pbds::detail::pat_trie_base::_Inode< ATraits, Metadata >::iterator
```

Child iterator.

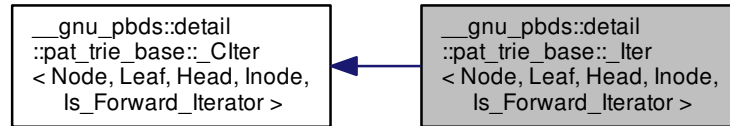
Definition at line 320 of file pat_trie_base.hpp.

The documentation for this struct was generated from the following file:

- [pat_trie_base.hpp](#)

5.295 `__gnu_pbds::detail::pat_trie_base::_Iter< Node, Leaf, Head, Inode, Is_Forward_Iterator >` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::pat_trie_base::_Iter< Node, Leaf, Head, Inode, Is_Forward_Iterator >`:



Public Types

- `typedef _Alloc::template rebind< Head > __rebind_h`
- `typedef _Alloc::template rebind< Inode > __rebind_in`
- `typedef _Alloc::template rebind< Leaf > __rebind_l`
- `typedef _Alloc::template rebind< Node > __rebind_n`
- `typedef allocator_type _Alloc`
- `typedef base_type::allocator_type allocator_type`
- `typedef _CIter< Node, Leaf, Head, Inode, Is_Forward_Iterator > base_type`
- `typedef type_traits::const_pointer const_pointer`
- `typedef type_traits::const_reference const_reference`
- `typedef allocator_type::difference_type difference_type`
- `typedef base_type::head_pointer head_pointer`
- `typedef Inode::iterator inode_iterator`
- `typedef base_type::inode_pointer inode_pointer`
- `typedef std::bidirectional_iterator_tag iterator_category`
- `typedef base_type::leaf_const_pointer leaf_const_pointer`
- `typedef base_type::leaf_pointer leaf_pointer`
- `typedef base_type::node_pointer node_pointer`
- `typedef type_traits::pointer pointer`
- `typedef type_traits::reference reference`
- `typedef base_type::type_traits type_traits`
- `typedef type_traits::value_type value_type`

Public Member Functions

- `_Iter (node_pointer p_nd=0)`
- `_Iter (const _Iter< Node, Leaf, Head, Inode,!Is_Forward_Iterator > &other)`
- `bool operator!= (const _CIter &other) const`
- `bool operator!= (const _CIter< Node, Leaf, Head, Inode,!Is_Forward_Iterator > &other) const`
- `reference operator* () const`
- `_Iter & operator++ ()`

- [_Iter](#) **operator++** (int)
- [_Iter](#) & **operator--** ()
- [_Iter](#) **operator--** (int)
- pointer **operator->** () const
- [_Iter](#) & **operator=** (const [_Iter](#) &other)
- [_Iter](#) & **operator=** (const [_Iter](#)< Node, Leaf, Head, Inode, !Is_Forward_Iterator > &other)
- bool **operator==** (const [_CIter](#) &other) const
- bool **operator==** (const [_CIter](#)< Node, Leaf, Head, Inode, !Is_Forward_Iterator > &other) const

Public Attributes

- node_pointer **m_p_nd**

Protected Member Functions

- void **dec** (false_type)
- void **dec** (true_type)
- void **inc** (false_type)
- void **inc** (true_type)

Static Protected Member Functions

- static node_pointer **get_larger_sibling** (node_pointer p_nd)
- static node_pointer **get_smaller_sibling** (node_pointer p_nd)
- static leaf_pointer **leftmost_descendant** (node_pointer p_nd)
- static leaf_pointer **rightmost_descendant** (node_pointer p_nd)

5.295.1 Detailed Description

```
template<typename Node, typename Leaf, typename Head, typename Inode, bool Is_Forward_Iterator>
class __gnu_pbds::detail::pat_trie_base::_Iter< Node, Leaf, Head, Inode, Is_Forward_Iterator >
```

Iterator.

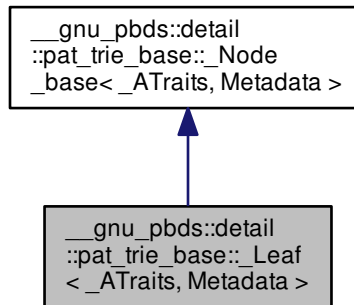
Definition at line 713 of file pat_trie_base.hpp.

The documentation for this class was generated from the following file:

- [pat_trie_base.hpp](#)

5.296 `__gnu_pbds::detail::pat_trie_base::_Leaf<_ATraits, Metadata>` Struct Template Reference

Inheritance diagram for `__gnu_pbds::detail::pat_trie_base::_Leaf<_ATraits, Metadata>`:



Public Types

- typedef `_Alloc::template rebind<_ATraits>` **__rebind_at**
- typedef `_Alloc::template rebind<_Node_base>` **__rebind_n**
- typedef `_ATraits::const_iterator` **a_const_iterator**
- typedef `__rebind_at::other::const_pointer` **a_const_pointer**
- typedef `_ATraits` **access_traits**
- typedef `_Alloc` **allocator_type**
- typedef `_Node_base<_ATraits, Metadata>` **base_type**
- typedef `type_traits::const_reference` **const_reference**
- typedef `__rebind_n::other::pointer` **node_pointer**
- typedef `type_traits::reference` **reference**
- typedef `base_type::type_traits` **type_traits**
- typedef `type_traits::value_type` **value_type**

Public Member Functions

- **_Leaf** (const_reference other)
- reference **value** ()
- const_reference **value** () const

Public Attributes

- node_pointer **m_p_parent**
- const [node_type](#) **m_type**

5.296.1 Detailed Description

```
template<typename ATraits, typename Metadata>
struct __gnu_pbds::detail::pat_trie_base::_Leaf< ATraits, Metadata >
```

Leaf node for PATRICIA tree.

Definition at line 162 of file pat_trie_base.hpp.

The documentation for this struct was generated from the following file:

- [pat_trie_base.hpp](#)

5.297 __gnu_pbds::detail::pat_trie_base::_Metadata< Metadata, _Alloc > Struct Template Reference

Public Types

- typedef _Alloc::template rebind< Metadata > __**rebind_m**
- typedef _Alloc **allocator_type**
- typedef __rebind_m::other::const_reference **const_reference**
- typedef Metadata **metadata_type**

Public Member Functions

- const_reference **get_metadata** () const

Public Attributes

- metadata_type **m_metadata**

5.297.1 Detailed Description

```
template<typename Metadata, typename _Alloc>
struct __gnu_pbds::detail::pat_trie_base::_Metadata< Metadata, _Alloc >
```

Metadata base primary template.

Definition at line 67 of file pat_trie_base.hpp.

The documentation for this struct was generated from the following file:

- [pat_trie_base.hpp](#)

5.298 `__gnu_pbds::detail::pat_trie_base::_Metadata< null_type, _Alloc >` Struct Template Reference

Public Types

- typedef `_Alloc` **allocator_type**
- typedef `null_type` **metadata_type**

5.298.1 Detailed Description

```
template<typename _Alloc>
struct __gnu_pbds::detail::pat_trie_base::_Metadata< null_type, _Alloc >
```

Specialization for null metadata.

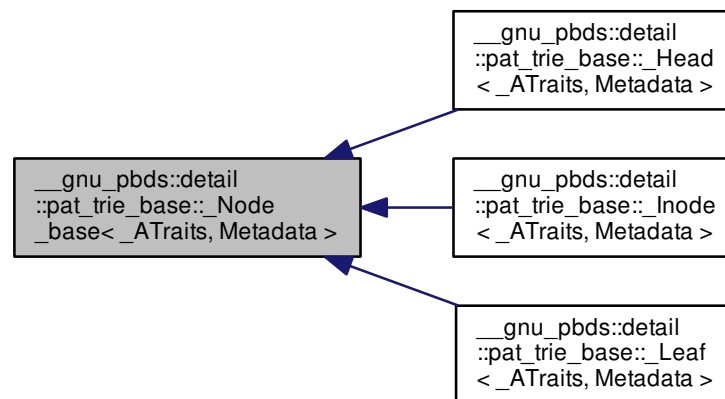
Definition at line 83 of file `pat_trie_base.hpp`.

The documentation for this struct was generated from the following file:

- [pat_trie_base.hpp](#)

5.299 `__gnu_pbds::detail::pat_trie_base::_Node_base< _ATraits, Metadata >` Struct Template Reference

Inheritance diagram for `__gnu_pbds::detail::pat_trie_base::_Node_base< _ATraits, Metadata >`:



Public Types

- typedef __Alloc::template rebind< __ATraits > **__rebind_at**
- typedef __Alloc::template rebind< [__Node_base](#) > **__rebind_n**
- typedef __ATraits::const_iterator **a_const_iterator**
- typedef __rebind_at::other::const_pointer **a_const_pointer**
- typedef __ATraits **access_traits**
- typedef __Alloc **allocator_type**
- typedef __rebind_n::other::pointer **node_pointer**
- typedef __ATraits::type_traits **type_traits**

Public Member Functions

- [__Node_base](#) ([node_type](#) type)

Public Attributes

- node_pointer **m_p_parent**
- const [node_type](#) **m_type**

5.299.1 Detailed Description

```
template<typename __ATraits, typename Metadata>
struct __gnu_pbds::detail::pat_trie_base::_Node_base< __ATraits, Metadata >
```

Node base.

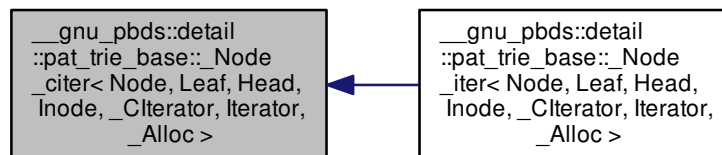
Definition at line 92 of file pat_trie_base.hpp.

The documentation for this struct was generated from the following file:

- [pat_trie_base.hpp](#)

5.300 __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, __CIterator, Iterator, __Alloc > Class Template Reference

Inheritance diagram for __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, __CIterator, Iterator, __Alloc >:



Public Types

- typedef `_Alloc::template rebind< metadata_type > __rebind_m`
- typedef `__rebind_m::other __rebind_ma`
- typedef `value_type const_reference`
- typedef `trivial_iterator_difference_type difference_type`
- typedef `trivial_iterator_tag iterator_category`
- typedef `__rebind_ma::const_reference metadata_const_reference`
- typedef `Node::metadata_type metadata_type`
- typedef `value_type reference`
- typedef `_Alloc::size_type size_type`
- typedef `_Citerator value_type`

Public Member Functions

- `_Node_citer (node_pointer p_nd=0, a_const_pointer p_traits=0)`
- `_Node_citer get_child (size_type i) const`
- `metadata_const_reference get_metadata () const`
- `size_type num_children () const`
- `bool operator!= (const _Node_citer &other) const`
- `const_reference operator* () const`
- `bool operator== (const _Node_citer &other) const`
- `std::pair< a_const_iterator, a_const_iterator > valid_prefix () const`

Public Attributes

- node_pointer **m_p_nd**
- a_const_pointer **m_p_traits**

Protected Types

- typedef `_Alloc::template rebind< Inode > __rebind_in`
- typedef `_Alloc::template rebind< Leaf > __rebind_l`
- typedef `_Alloc::template rebind< Node > __rebind_n`
- typedef `Node::a_const_iterator a_const_iterator`
- typedef `Node::a_const_pointer a_const_pointer`
- typedef `__rebind_in::other::const_pointer inode_const_pointer`
- typedef `__rebind_in::other::pointer inode_pointer`
- typedef `__rebind_l::other::const_pointer leaf_const_pointer`
- typedef `__rebind_l::other::pointer leaf_pointer`
- typedef `__rebind_n::other::pointer node_pointer`

5.300.1 Detailed Description

```
template<typename Node, typename Leaf, typename Head, typename Inode, typename _Citerator, typename Iterator, typename _↵
Alloc>
class __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _Citerator, Iterator, _Alloc >
```

Node const iterator.

Definition at line 814 of file `pat_trie_base.hpp`.

5.300.2 Member Typedef Documentation

5.300.2.1 `template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator , typename Iterator , typename _Alloc > typedef _Alloc::template rebind<metadata_type> __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::_rebind_m`

Const metadata reference type.

Definition at line 869 of file `pat_trie_base.hpp`.

5.300.2.2 `template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator , typename Iterator , typename _Alloc > typedef Node::metadata_type __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::_metadata_type`

Metadata type.

Definition at line 866 of file `pat_trie_base.hpp`.

5.300.3 Member Function Documentation

5.300.3.1 `template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator , typename Iterator , typename _Alloc > _Node_citer __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::get_child (size_type i) const [inline]`

Returns a `__const` node `__iterator` to the corresponding node's `i`-th child.

Definition at line 911 of file `pat_trie_base.hpp`.

References `std::advance()`.

5.300.3.2 `template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator , typename Iterator , typename _Alloc > metadata_const_reference __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::get_metadata () const [inline]`

Metadata access.

Definition at line 894 of file `pat_trie_base.hpp`.

5.300.3.3 `template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator , typename Iterator , typename _Alloc > size_type __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::num_children () const [inline]`

Returns the number of children in the corresponding node.

Definition at line 899 of file `pat_trie_base.hpp`.

References `std::distance()`.

5.300.3.4 `template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator , typename Iterator ,
typename _Alloc > bool __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator,
Iterator, _Alloc >::operator!= (const _Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc > & other)
const [inline]`

Compares content (negatively) to a different iterator object.

Definition at line 927 of file `pat_trie_base.hpp`.

5.300.3.5 `template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator , typename Iterator ,
typename _Alloc > const_reference __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode,
_CIterator, Iterator, _Alloc >::operator* () const [inline]`

Const access; returns the `__const_iterator*` associated with the current leaf.

Definition at line 886 of file `pat_trie_base.hpp`.

5.300.3.6 `template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator , typename Iterator ,
typename _Alloc > bool __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator,
Iterator, _Alloc >::operator== (const _Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc > & other)
const [inline]`

Compares content to a different iterator object.

Definition at line 922 of file `pat_trie_base.hpp`.

5.300.3.7 `template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator , typename
Iterator , typename _Alloc > std::pair<a_const_iterator, a_const_iterator> __gnu_pbds::detail::pat_←
trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::valid_prefix () const
[inline]`

Subtree valid prefix.

Definition at line 880 of file `pat_trie_base.hpp`.

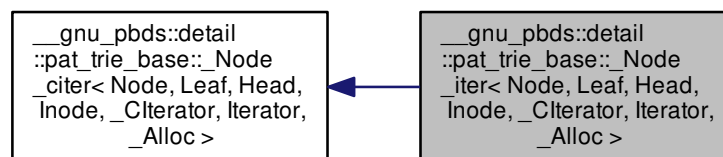
References `std::make_pair()`.

The documentation for this class was generated from the following file:

- [pat_trie_base.hpp](#)

5.301 `__gnu_pbds::detail::pat_trie_base::_Node_iter< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::pat_trie_base::_Node_iter< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >`:



Public Types

- typedef `_Alloc::template rebind< metadata_type > __rebind_m`
- typedef `__rebind_m::other __rebind_ma`
- typedef `value_type const_reference`
- typedef `trivial_iterator_difference_type difference_type`
- typedef `trivial_iterator_tag iterator_category`
- typedef `__rebind_ma::const_reference metadata_const_reference`
- typedef `Node::metadata_type metadata_type`
- typedef `value_type reference`
- typedef `base_type::size_type size_type`
- typedef `Iterator value_type`

Public Member Functions

- `_Node_iter (node_pointer p_nd=0, a_const_pointer p_traits=0)`
- `_Node_iter get_child (size_type i) const`
- `metadata_const_reference get_metadata () const`
- `size_type num_children () const`
- `bool operator!= (const _Node_citer &other) const`
- `reference operator* () const`
- `bool operator== (const _Node_citer &other) const`
- `std::pair< a_const_iterator, a_const_iterator > valid_prefix () const`

Public Attributes

- node_pointer **m_p_nd**
- a_const_pointer **m_p_traits**

Protected Types

- typedef `_Alloc::template rebind< Inode > __rebind_in`
- typedef `_Alloc::template rebind< Leaf > __rebind_l`
- typedef `Node::a_const_iterator a_const_iterator`
- typedef `__rebind_in::other::const_pointer inode_const_pointer`
- typedef `__rebind_l::other::const_pointer leaf_const_pointer`
- typedef `__rebind_l::other::pointer leaf_pointer`

5.301.1 Detailed Description

```
template<typename Node, typename Leaf, typename Head, typename Inode, typename _Citerator, typename Iterator, typename _C↵
Alloc>
class __gnu_pbds::detail::pat_trie_base::_Node_iter< Node, Leaf, Head, Inode, _Citerator, Iterator, _Alloc >
```

Node iterator.

Definition at line 943 of file `pat_trie_base.hpp`.

5.301.2 Member Typedef Documentation

5.301.2.1 `template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator , typename Iterator , typename _Alloc > typedef _Alloc::template rebind<metadata_type> __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::_rebind_m [inherited]`

Const metadata reference type.

Definition at line 869 of file `pat_trie_base.hpp`.

5.301.2.2 `template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator , typename Iterator , typename _Alloc > typedef Node::metadata_type __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::metadata_type [inherited]`

Metadata type.

Definition at line 866 of file `pat_trie_base.hpp`.

5.301.3 Member Function Documentation

5.301.3.1 `template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator , typename Iterator , typename _Alloc > _Node_iter __gnu_pbds::detail::pat_trie_base::_Node_iter< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::get_child (size_type i) const [inline]`

Returns a node __iterator to the corresponding node's i-th child.

Definition at line 976 of file `pat_trie_base.hpp`.

References `std::advance()`, `std::begin()`, `std::distance()`, `std::end()`, and `std::make_pair()`.

5.301.3.2 `template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator , typename Iterator , typename _Alloc > metadata_const_reference __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::get_metadata () const [inline],[inherited]`

Metadata access.

Definition at line 894 of file `pat_trie_base.hpp`.

5.301.3.3 `template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator , typename Iterator , typename _Alloc > size_type __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::num_children () const [inline],[inherited]`

Returns the number of children in the corresponding node.

Definition at line 899 of file `pat_trie_base.hpp`.

References `std::distance()`.

5.301.3.4 `template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator , typename Iterator ,
typename _Alloc > bool __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator,
Iterator, _Alloc >::operator!=(const _Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc > & other)
const [inline],[inherited]`

Compares content (negatively) to a different iterator object.

Definition at line 927 of file pat_trie_base.hpp.

5.301.3.5 `template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator , typename Iterator ,
typename _Alloc > reference __gnu_pbds::detail::pat_trie_base::_Node_iter< Node, Leaf, Head, Inode,
_CIterator, Iterator, _Alloc >::operator*() const [inline]`

Access; returns the iterator* associated with the current leaf.

Definition at line 968 of file pat_trie_base.hpp.

5.301.3.6 `template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator , typename Iterator ,
typename _Alloc > bool __gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _CIterator,
Iterator, _Alloc >::operator==(const _Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc > & other)
const [inline],[inherited]`

Compares content to a different iterator object.

Definition at line 922 of file pat_trie_base.hpp.

5.301.3.7 `template<typename Node , typename Leaf , typename Head , typename Inode , typename _CIterator , typename
Iterator , typename _Alloc > std::pair<a_const_iterator, a_const_iterator> __gnu_pbds::detail::pat_trie_↵
base::_Node_citer< Node, Leaf, Head, Inode, _CIterator, Iterator, _Alloc >::valid_prefix () const [inline],
[inherited]`

Subtree valid prefix.

Definition at line 880 of file pat_trie_base.hpp.

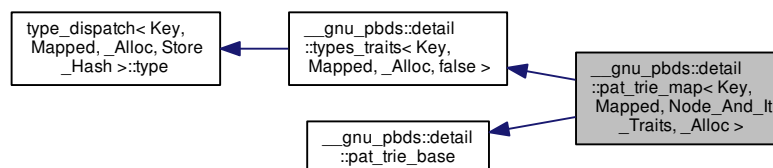
References `std::make_pair()`.

The documentation for this class was generated from the following file:

- [pat_trie_base.hpp](#)

5.302 `__gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >`:



Public Types

- typedef traits_type::access_traits **access_traits**
- typedef _Alloc **allocator_type**
- typedef [std::pair](#)< size_type, size_type > **comp_hash**
- typedef point_const_iterator **const_iterator**
- typedef traits_base::const_pointer **const_pointer**
- typedef traits_base::const_reference **const_reference**
- typedef traits_type::const_reverse_iterator **const_reverse_iterator**
- typedef [pat_trie_tag](#) **container_category**
- typedef _Alloc::difference_type **difference_type**
- typedef point_iterator **iterator**
- typedef traits_base::key_const_pointer **key_const_pointer**
- typedef traits_base::key_const_reference **key_const_reference**
- typedef traits_base::key_pointer **key_pointer**
- typedef traits_base::key_reference **key_reference**
- typedef traits_base::key_type **key_type**
- typedef traits_base::mapped_const_pointer **mapped_const_pointer**
- typedef traits_base::mapped_const_reference **mapped_const_reference**
- typedef traits_base::mapped_pointer **mapped_pointer**
- typedef traits_base::mapped_reference **mapped_reference**
- typedef traits_base::mapped_type **mapped_type**
- typedef __nothrowcopy::indicator **no_throw_indicator**
- typedef traits_type::node_const_iterator **node_const_iterator**
- typedef traits_type::node_iterator **node_iterator**
- enum [node_type](#) { **i_node**, **leaf_node**, **head_node** }
- typedef traits_type::node_update **node_update**
- typedef traits_type::const_iterator **point_const_iterator**
- typedef traits_type::iterator **point_iterator**
- typedef traits_base::pointer **pointer**
- typedef traits_base::reference **reference**
- typedef traits_type::reverse_iterator **reverse_iterator**
- typedef _Alloc::size_type **size_type**
- typedef integral_constant< int, Store_Hash > **store_extra**
- typedef traits_base::value_type **value_type**

Public Member Functions

- **pat_trie_map** (const access_traits &)
- **pat_trie_map** (const [pat_trie_map](#)< Key, Mapped, Node_And_It_Traits, _Alloc > &)
- iterator **begin** ()
- const_iterator **begin** () const
- void **clear** ()
- bool **empty** () const
- iterator **end** ()
- const_iterator **end** () const
- bool **erase** (key_const_reference)
- const_iterator **erase** (const_iterator)
- iterator **erase** (iterator)
- const_reverse_iterator **erase** (const_reverse_iterator)

- reverse_iterator **erase** (reverse_iterator)
- template<typename Pred >
size_type **erase_if** (Pred)
- point_iterator **find** (key_const_reference)
- point_const_iterator **find** (key_const_reference) const
- access_traits & **get_access_traits** ()
- const access_traits & **get_access_traits** () const
- node_update & **get_node_update** ()
- const node_update & **get_node_update** () const
- [std::pair](#)< point_iterator, bool > **insert** (const_reference)
- void **join** ([pat_trie_map](#)< Key, Mapped, Node_And_It_Traits, _Alloc > &)
- point_iterator **lower_bound** (key_const_reference)
- point_const_iterator **lower_bound** (key_const_reference) const
- size_type **max_size** () const
- node_const_iterator [node_begin](#) () const
- node_iterator [node_begin](#) ()
- node_const_iterator [node_end](#) () const
- node_iterator [node_end](#) ()
- mapped_reference **operator[]** (key_const_reference r_key)
- reverse_iterator **rbegin** ()
- const_reverse_iterator **rbegin** () const
- reverse_iterator **rend** ()
- const_reverse_iterator **rend** () const
- size_type **size** () const
- void **split** (key_const_reference, [pat_trie_map](#)< Key, Mapped, Node_And_It_Traits, _Alloc > &)
- void **swap** ([pat_trie_map](#)< Key, Mapped, Node_And_It_Traits, _Alloc > &)
- point_iterator **upper_bound** (key_const_reference)
- point_const_iterator **upper_bound** (key_const_reference) const

Public Attributes

- no_throw_indicator **m_no_throw_copies_indicator**
- store_extra **m_store_extra_indicator**

Protected Member Functions

- template<typename It >
void **copy_from_range** (It, It)
- node_pointer **recursive_copy_node** (node_const_pointer)
- void **value_swap** ([pat_trie_map](#)< Key, Mapped, Node_And_It_Traits, _Alloc > &)

5.302.1 Detailed Description

```
template<typename Key, typename Mapped, typename Node_And_It_Traits, typename _Alloc>
class __gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >
```

PATRICIA trie.

This implementation loosely borrows ideas from: 1) Fast Mergeable Integer Maps, Okasaki, Gill 1998 2) Ptsset: Sets of integers implemented as Patricia trees, Jean-Christophe Filliatr, 2000.

Definition at line 101 of file pat_trie_.hpp.

5.302.2 Member Enumeration Documentation

5.302.2.1 `enum __gnu_pbds::detail::pat_trie_base::node_type` [inherited]

Three types of nodes.

`i_node` is used by `_Inode`, `leaf_node` by `_Leaf`, and `head_node` by `_Head`.

Definition at line 58 of file `pat_trie_base.hpp`.

5.302.3 Member Function Documentation

5.302.3.1 `template<typename Key , typename Mapped , typename Node_And_It_Traits , typename _Alloc > pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >::node_const_iterator __gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >::node_begin () const` [inline]

Returns a `const node_iterator` corresponding to the node at the root of the tree.

Definition at line 101 of file `pat_trie_.hpp`.

5.302.3.2 `template<typename Key , typename Mapped , typename Node_And_It_Traits , typename _Alloc > pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >::node_iterator __gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >::node_begin ()` [inline]

Returns a `node_iterator` corresponding to the node at the root of the tree.

Definition at line 107 of file `pat_trie_.hpp`.

5.302.3.3 `template<typename Key , typename Mapped , typename Node_And_It_Traits , typename _Alloc > pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >::node_const_iterator __gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >::node_end () const` [inline]

Returns a `const node_iterator` corresponding to a node just after a leaf of the tree.

Definition at line 113 of file `pat_trie_.hpp`.

5.302.3.4 `template<typename Key , typename Mapped , typename Node_And_It_Traits , typename _Alloc > pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >::node_iterator __gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >::node_end ()` [inline]

Returns a `node_iterator` corresponding to a node just after a leaf of the tree.

Definition at line 119 of file `pat_trie_.hpp`.

References `std::begin()`, `std::end()`, `std::pair< _T1, _T2 >::first`, `std::make_pair()`, `__gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >::node_end()`, `std::rbegin()`, and `std::rend()`.

Referenced by `__gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >::node_end()`.

The documentation for this class was generated from the following file:

- [pat_trie_.hpp](#)

5.303 `__gnu_pbds::detail::probe_fn_base<_Alloc>` Class Template Reference

5.303.1 Detailed Description

```
template<typename _Alloc>
class __gnu_pbds::detail::probe_fn_base<_Alloc>
```

Probe functor base.

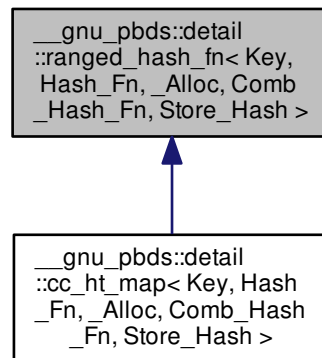
Definition at line 52 of file `probe_fn_base.hpp`.

The documentation for this class was generated from the following file:

- [probe_fn_base.hpp](#)

5.304 `__gnu_pbds::detail::ranged_hash_fn<Key, Hash_Fn, _Alloc, Comb_Hash_Fn, Store_Hash>` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::ranged_hash_fn<Key, Hash_Fn, _Alloc, Comb_Hash_Fn, Store_Hash>`:



5.304.1 Detailed Description

```
template<typename Key, typename Hash_Fn, typename _Alloc, typename Comb_Hash_Fn, bool Store_Hash>
class __gnu_pbds::detail::ranged_hash_fn<Key, Hash_Fn, _Alloc, Comb_Hash_Fn, Store_Hash>
```

Primary template.

Definition at line 55 of file `ranged_hash_fn.hpp`.

The documentation for this class was generated from the following file:

- [ranged_hash_fn.hpp](#)

5.305 `__gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, false >` Class Template Reference

Inherits Hash_Fn, and Comb_Hash_Fn.

Protected Types

- typedef Comb_Hash_Fn **comb_hash_fn_base**
- typedef Hash_Fn **hash_fn_base**
- typedef _Alloc::template rebind< Key >::other **key_allocator**
- typedef key_allocator::const_reference **key_const_reference**
- typedef _Alloc::size_type **size_type**

Protected Member Functions

- **ranged_hash_fn** (size_type)
- **ranged_hash_fn** (size_type, const Hash_Fn &)
- **ranged_hash_fn** (size_type, const Hash_Fn &, const Comb_Hash_Fn &)
- void **notify_resized** (size_type)
- size_type **operator()** (key_const_reference) const
- void **swap** ([ranged_hash_fn](#)< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, false > &)

5.305.1 Detailed Description

```
template<typename Key, typename Hash_Fn, typename _Alloc, typename Comb_Hash_Fn>
class __gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, false >
```

Specialization 1 The client supplies a hash function and a ranged hash function, and requests that hash values not be stored.

Definition at line 71 of file [ranged_hash_fn.hpp](#).

The documentation for this class was generated from the following file:

- [ranged_hash_fn.hpp](#)

5.306 `__gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, true >` Class Template Reference

Inherits Hash_Fn, and Comb_Hash_Fn.

Protected Types

- typedef Comb_Hash_Fn **comb_hash_fn_base**
- typedef [std::pair](#)< size_type, size_type > **comp_hash**
- typedef Hash_Fn **hash_fn_base**
- typedef _Alloc::template rebind< Key >::other **key_allocator**
- typedef key_allocator::const_reference **key_const_reference**
- typedef _Alloc::size_type **size_type**

Protected Member Functions

- **ranged_hash_fn** (size_type)
- **ranged_hash_fn** (size_type, const Hash_Fn &)
- **ranged_hash_fn** (size_type, const Hash_Fn &, const Comb_Hash_Fn &)
- void **notify_resized** (size_type)
- [comp_hash operator\(\)](#) (key_const_reference) const
- [comp_hash operator\(\)](#) (key_const_reference, size_type) const
- void **swap** ([ranged_hash_fn](#)< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, true > &)

5.306.1 Detailed Description

```
template<typename Key, typename Hash_Fn, typename _Alloc, typename Comb_Hash_Fn>
class __gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, true >
```

Specialization 2 The client supplies a hash function and a ranged hash function, and requests that hash values be stored.

Definition at line 153 of file ranged_hash_fn.hpp.

The documentation for this class was generated from the following file:

- [ranged_hash_fn.hpp](#)

5.307 __gnu_pbds::detail::ranged_hash_fn< Key, null_type, _Alloc, Comb_Hash_Fn, false > Class Template Reference

Inherits Comb_Hash_Fn.

Protected Types

- typedef Comb_Hash_Fn **comb_hash_fn_base**
- typedef _Alloc::size_type **size_type**

Protected Member Functions

- **ranged_hash_fn** (size_type)
- **ranged_hash_fn** (size_type, const Comb_Hash_Fn &)
- **ranged_hash_fn** (size_type, const [null_type](#) &, const Comb_Hash_Fn &)
- void **swap** ([ranged_hash_fn](#)< Key, [null_type](#), _Alloc, Comb_Hash_Fn, false > &)

5.307.1 Detailed Description

```
template<typename Key, typename _Alloc, typename Comb_Hash_Fn>
class __gnu_pbds::detail::ranged_hash_fn< Key, null_type, _Alloc, Comb_Hash_Fn, false >
```

Specialization 3 The client does not supply a hash function (by specifying `null_type` as the `Hash_Fn` parameter), and requests that hash values not be stored.

Definition at line 255 of file `ranged_hash_fn.hpp`.

The documentation for this class was generated from the following file:

- [ranged_hash_fn.hpp](#)

5.308 `__gnu_pbds::detail::ranged_hash_fn< Key, null_type, _Alloc, Comb_Hash_Fn, true >` Class Template Reference

Inherits `Comb_Hash_Fn`.

Protected Types

- typedef `Comb_Hash_Fn` **comb_hash_fn_base**
- typedef `_Alloc::size_type` **size_type**

Protected Member Functions

- **ranged_hash_fn** (size_type)
- **ranged_hash_fn** (size_type, const Comb_Hash_Fn &)
- **ranged_hash_fn** (size_type, const [null_type](#) &, const Comb_Hash_Fn &)
- void **swap** ([ranged_hash_fn](#)< Key, [null_type](#), _Alloc, Comb_Hash_Fn, true > &)

5.308.1 Detailed Description

```
template<typename Key, typename _Alloc, typename Comb_Hash_Fn>
class __gnu_pbds::detail::ranged_hash_fn< Key, null_type, _Alloc, Comb_Hash_Fn, true >
```

Specialization 4 The client does not supply a hash function (by specifying `null_type` as the `Hash_Fn` parameter), and requests that hash values be stored.

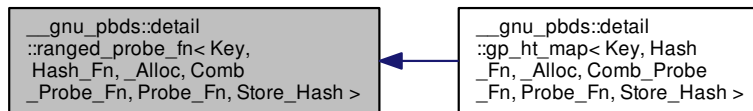
Definition at line 312 of file `ranged_hash_fn.hpp`.

The documentation for this class was generated from the following file:

- [ranged_hash_fn.hpp](#)

5.309 `__gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, Store_Hash >` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, Store_Hash >`:



5.309.1 Detailed Description

```
template<typename Key, typename Hash_Fn, typename _Alloc, typename Comb_Probe_Fn, typename Probe_Fn, bool Store_Hash>
class __gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, Store_Hash >
```

Primary template.

Definition at line 55 of file `ranged_probe_fn.hpp`.

The documentation for this class was generated from the following file:

- [ranged_probe_fn.hpp](#)

5.310 `__gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, false >` Class Template Reference

Inherits `Hash_Fn`, `Comb_Probe_Fn`, and `Probe_Fn`.

Protected Types

- typedef `Comb_Probe_Fn` **comb_probe_fn_base**
- typedef `Hash_Fn` **hash_fn_base**
- typedef `_Alloc::template rebind< Key >::other` **key_allocator**
- typedef `key_allocator::const_reference` **key_const_reference**
- typedef `Probe_Fn` **probe_fn_base**
- typedef `_Alloc::size_type` **size_type**

Protected Member Functions

- **ranged_probe_fn** (size_type)
- **ranged_probe_fn** (size_type, const Hash_Fn &)
- **ranged_probe_fn** (size_type, const Hash_Fn &, const Comb_Probe_Fn &)
- **ranged_probe_fn** (size_type, const Hash_Fn &, const Comb_Probe_Fn &, const Probe_Fn &)
- void **notify_resized** (size_type)
- size_type **operator()** (key_const_reference) const
- size_type **operator()** (key_const_reference, size_type, size_type) const
- void **swap** ([ranged_probe_fn](#)< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, false > &)

5.310.1 Detailed Description

```
template<typename Key, typename Hash_Fn, typename _Alloc, typename Comb_Probe_Fn, typename Probe_Fn>
class __gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, false >
```

Specialization 1 The client supplies a probe function and a ranged probe function, and requests that hash values not be stored.

Definition at line 71 of file `ranged_probe_fn.hpp`.

The documentation for this class was generated from the following file:

- [ranged_probe_fn.hpp](#)

5.311 `__gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, true >` Class Template Reference

Inherits `Hash_Fn`, `Comb_Probe_Fn`, and `Probe_Fn`.

Protected Types

- typedef `Comb_Probe_Fn` **comb_probe_fn_base**
- typedef `std::pair< size_type, size_type >` **comp_hash**
- typedef `Hash_Fn` **hash_fn_base**
- typedef `_Alloc::template rebind< Key >::other` **key_allocator**
- typedef `key_allocator::const_reference` **key_const_reference**
- typedef `Probe_Fn` **probe_fn_base**
- typedef `_Alloc::size_type` **size_type**

Protected Member Functions

- **ranged_probe_fn** (size_type)
- **ranged_probe_fn** (size_type, const Hash_Fn &)
- **ranged_probe_fn** (size_type, const Hash_Fn &, const Comb_Probe_Fn &)
- **ranged_probe_fn** (size_type, const Hash_Fn &, const Comb_Probe_Fn &, const Probe_Fn &)
- void **notify_resized** (size_type)
- [comp_hash](#) **operator()** (key_const_reference) const
- size_type **operator()** (key_const_reference, size_type, size_type) const
- size_type **operator()** (key_const_reference, size_type) const
- void **swap** ([ranged_probe_fn](#)< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, true > &)

5.311.1 Detailed Description

```
template<typename Key, typename Hash_Fn, typename _Alloc, typename Comb_Probe_Fn, typename Probe_Fn>
class __gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, true >
```

Specialization 2- The client supplies a probe function and a ranged probe function, and requests that hash values not be stored.

Definition at line 176 of file `ranged_probe_fn.hpp`.

The documentation for this class was generated from the following file:

- [ranged_probe_fn.hpp](#)

5.312 __gnu_pbds::detail::ranged_probe_fn< Key, null_type, _Alloc, Comb_Probe_Fn, null_type, false > Class Template Reference

Inherits `Comb_Probe_Fn`.

Protected Types

- typedef `Comb_Probe_Fn` **comb_probe_fn_base**
- typedef `_Alloc::template rebind< Key >::other` **key_allocator**
- typedef `key_allocator::const_reference` **key_const_reference**
- typedef `_Alloc::size_type` **size_type**

Protected Member Functions

- **ranged_probe_fn** (`size_type` size)
- **ranged_probe_fn** (`size_type`, const `Comb_Probe_Fn` &r_comb_probe_fn)
- **ranged_probe_fn** (`size_type`, const [null_type](#) &, const `Comb_Probe_Fn` &r_comb_probe_fn, const [null_type](#) &)
- void **swap** ([ranged_probe_fn](#) &other)

5.312.1 Detailed Description

```
template<typename Key, typename _Alloc, typename Comb_Probe_Fn>
class __gnu_pbds::detail::ranged_probe_fn< Key, null_type, _Alloc, Comb_Probe_Fn, null_type, false >
```

Specialization 3 and 4 The client does not supply a hash function or probe function, and requests that hash values not be stored.

Definition at line 296 of file `ranged_probe_fn.hpp`.

The documentation for this class was generated from the following file:

- [ranged_probe_fn.hpp](#)

5.313 `__gnu_pbds::detail::rb_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >` Class Template Reference

Inherits `__gnu_pbds::detail::bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >`.

Public Types

- typedef `_Alloc` **allocator_type**
- typedef `Cmp_Fn` **cmp_fn**
- typedef `std::pair< size_type, size_type >` **comp_hash**
- typedef `base_type::const_iterator` **const_iterator**
- typedef `base_type::const_pointer` **const_pointer**
- typedef `base_type::const_reference` **const_reference**
- typedef `base_type::const_reverse_iterator` **const_reverse_iterator**
- typedef `rb_tree_tag` **container_category**
- typedef `_Alloc::difference_type` **difference_type**
- typedef `base_type::iterator` **iterator**
- typedef `base_type::key_const_pointer` **key_const_pointer**
- typedef `base_type::key_const_reference` **key_const_reference**
- typedef `base_type::key_pointer` **key_pointer**
- typedef `base_type::key_reference` **key_reference**
- typedef `base_type::key_type` **key_type**
- typedef `base_type::mapped_const_pointer` **mapped_const_pointer**
- typedef `base_type::mapped_const_reference` **mapped_const_reference**
- typedef `base_type::mapped_pointer` **mapped_pointer**
- typedef `base_type::mapped_reference` **mapped_reference**
- typedef `base_type::mapped_type` **mapped_type**
- typedef `__nothrowcopy::indicator` **no_throw_indicator**
- typedef `traits_type::node_const_iterator` **node_const_iterator**
- typedef `traits_type::node_iterator` **node_iterator**
- typedef `base_type::node_update` **node_update**
- typedef `base_type::const_iterator` **point_const_iterator**
- typedef `base_type::point_iterator` **point_iterator**
- typedef `base_type::pointer` **pointer**
- typedef `base_type::reference` **reference**
- typedef `base_type::reverse_iterator` **reverse_iterator**
- typedef `_Alloc::size_type` **size_type**
- typedef `integral_constant< int, Store_Hash >` **store_extra**
- typedef `base_type::value_type` **value_type**

Public Member Functions

- **rb_tree_map** (const `Cmp_Fn` &)
- **rb_tree_map** (const `Cmp_Fn` &, const `node_update` &)
- **rb_tree_map** (const `rb_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >` &)
- iterator **begin** ()
- const_iterator **begin** () const
- void **clear** ()

- `template<typename It >`
 `void copy_from_range (It, It)`
- `bool empty () const`
- `iterator end ()`
- `const_iterator end () const`
- `bool erase (key_const_reference)`
- `iterator erase (iterator)`
- `reverse_iterator erase (reverse_iterator)`
- `template<typename Pred >`
 `size_type erase_if (Pred)`
- `point_iterator find (key_const_reference)`
- `point_const_iterator find (key_const_reference) const`
- `Cmp_Fn & get_cmp_fn ()`
- `const Cmp_Fn & get_cmp_fn () const`
- `std::pair< point_iterator, bool > insert (const_reference)`
- `void join (rb_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)`
- `point_iterator lower_bound (key_const_reference)`
- `point_const_iterator lower_bound (key_const_reference) const`
- `size_type max_size () const`
- `node_const_iterator node_begin () const`
- `node_iterator node_begin ()`
- `node_const_iterator node_end () const`
- `node_iterator node_end ()`
- `mapped_reference operator[] (key_const_reference r_key)`
- `reverse_iterator rbegin ()`
- `const_reverse_iterator rbegin () const`
- `reverse_iterator rend ()`
- `const_reverse_iterator rend () const`
- `size_type size () const`
- `void split (key_const_reference, rb_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)`
- `void swap (rb_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)`
- `void swap (bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)`
- `point_iterator upper_bound (key_const_reference)`
- `point_const_iterator upper_bound (key_const_reference) const`

Public Attributes

- `no_throw_indicator m_no_throw_copies_indicator`
- `store_extra m_store_extra_indicator`

Protected Types

- `typedef node_allocator::value_type node`
- `typedef _Alloc::template rebind< typename traits_type::node >::other node_allocator`
- `typedef traits_type::null_node_update_pointer null_node_update_pointer`
- `typedef types_traits< Key, Mapped, _Alloc, false > traits_base`

Protected Member Functions

- void **actual_erase_node** (node_pointer)
- void **apply_update** (node_pointer, null_node_update_pointer)
- template<typename Node_Update_>
void **apply_update** (node_pointer, Node_Update_*)
- [std::pair](#)< node_pointer, bool > **erase** (node_pointer)
- node_pointer **get_new_node_for_leaf_insert** (const_reference, false_type)
- node_pointer **get_new_node_for_leaf_insert** (const_reference, true_type)
- void **initialize_min_max** ()
- iterator **insert_imp_empty** (const_reference)
- [std::pair](#)< point_iterator, bool > **insert_leaf** (const_reference)
- iterator **insert_leaf_new** (const_reference, node_pointer, bool)
- void **join_finish** (bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)
- bool **join_prep** (bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)
- size_type **recursive_count** (node_pointer) const
- void **rotate_left** (node_pointer)
- void **rotate_parent** (node_pointer)
- void **rotate_right** (node_pointer)
- void **split_finish** (bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)
- bool **split_prep** (key_const_reference, bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)
- void **update_min_max_for_erased_node** (node_pointer)
- void **update_to_top** (node_pointer, null_node_update_pointer)
- template<typename Node_Update_>
void **update_to_top** (node_pointer, Node_Update_*)
- void **value_swap** (bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)

Static Protected Member Functions

- static void **clear_imp** (node_pointer)

Protected Attributes

- node_pointer **m_p_head**
- size_type **m_size**

Static Protected Attributes

- static node_allocator **s_node_allocator**

5.313.1 Detailed Description

```
template<typename Key, typename Mapped, typename Cmp_Fn, typename Node_And_It_Traits, typename _Alloc>  
class __gnu_pbds::detail::rb_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >
```

Red-Black tree.

This implementation uses an idea from the SGI STL (using a *header* node which is needed for efficient iteration).

Definition at line 84 of file `rb_tree_.hpp`.

5.313.2 Member Function Documentation

5.313.2.1 `template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename _Alloc > bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_const_iterator
__gnu_pbds::detail::bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_begin () const
[inline], [inherited]`

Returns a const node_iterator corresponding to the node at the root of the tree.

Definition at line 109 of file bin_search_tree_.hpp.

5.313.2.2 `template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename _Alloc > bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_iterator
__gnu_pbds::detail::bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_begin ()
[inline], [inherited]`

Returns a node_iterator corresponding to the node at the root of the tree.

Definition at line 117 of file bin_search_tree_.hpp.

5.313.2.3 `template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename _Alloc > bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_const_iterator
__gnu_pbds::detail::bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_end () const
[inline], [inherited]`

Returns a const node_iterator corresponding to a node just after a leaf of the tree.

Definition at line 125 of file bin_search_tree_.hpp.

5.313.2.4 `template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename _Alloc > bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_iterator
__gnu_pbds::detail::bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_end ()
[inline], [inherited]`

Returns a node_iterator corresponding to a node just after a leaf of the tree.

Definition at line 133 of file bin_search_tree_.hpp.

The documentation for this class was generated from the following file:

- [rb_tree_.hpp](#)

5.314 __gnu_pbds::detail::rb_tree_node_< Value_Type, Metadata, _Alloc > Struct Template Reference

Public Types

- typedef _Alloc::template rebind< metadata_type >::other::const_reference **metadata_const_reference**
- typedef _Alloc::template rebind< metadata_type >::other::reference **metadata_reference**
- typedef Metadata **metadata_type**
- typedef _Alloc::template rebind< [rb_tree_node_< Value_Type, Metadata, _Alloc >](#) >::other::pointer **node_←
pointer**
- typedef Value_Type **value_type**

Public Member Functions

- `metadata_const_reference` **get_metadata** () const
- `metadata_reference` **get_metadata** ()
- `bool` **special** () const

Public Attributes

- `metadata_type` **m_metadata**
- `node_pointer` **m_p_left**
- `node_pointer` **m_p_parent**
- `node_pointer` **m_p_right**
- `bool` **m_red**
- `value_type` **m_value**

5.314.1 Detailed Description

```
template<typename Value_Type, class Metadata, typename _Alloc>
struct __gnu_pbds::detail::rb_tree_node_< Value_Type, Metadata, _Alloc >
```

Node for Red-Black trees.

Definition at line 52 of file `rb_tree_map_/node.hpp`.

The documentation for this struct was generated from the following file:

- [rb_tree_map_/node.hpp](#)

5.315 `__gnu_pbds::detail::rc< _Node, _Alloc >` Class Template Reference

Public Types

- `typedef entry_const_pointer` **const_iterator**
- `typedef node_pointer` **entry**

Public Member Functions

- `rc` (const [rc](#) &)
- `const const_iterator` **begin** () const
- `void` **clear** ()
- `bool` **empty** () const
- `const const_iterator` **end** () const
- `void` **pop** ()
- `void` **push** (entry)
- `size_type` **size** () const
- `void` **swap** ([rc](#) &)
- `node_pointer` **top** () const

5.315.1 Detailed Description

```
template<typename _Node, typename _Alloc>
class __gnu_pbds::detail::rc< _Node, _Alloc >
```

Redundant binary counter.

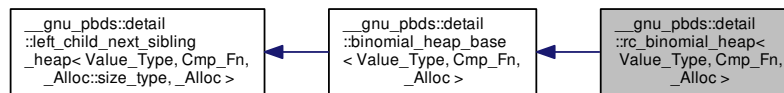
Definition at line 50 of file rc.hpp.

The documentation for this class was generated from the following file:

- [rc.hpp](#)

5.316 __gnu_pbds::detail::rc_binomial_heap< Value_Type, Cmp_Fn, _Alloc > Class Template Reference

Inheritance diagram for __gnu_pbds::detail::rc_binomial_heap< Value_Type, Cmp_Fn, _Alloc >:



Public Types

- typedef base_type::allocator_type **allocator_type**
- typedef base_type::cmp_fn **cmp_fn**
- typedef [base_type::const_iterator](#) **const_iterator**
- typedef base_type::const_pointer **const_pointer**
- typedef base_type::const_reference **const_reference**
- typedef _Alloc::difference_type **difference_type**
- typedef [base_type::iterator](#) **iterator**
- typedef [base_type::point_const_iterator](#) **point_const_iterator**
- typedef [base_type::point_iterator](#) **point_iterator**
- typedef base_type::pointer **pointer**
- typedef base_type::reference **reference**
- typedef _Alloc::size_type **size_type**
- typedef Value_Type **value_type**

Public Member Functions

- `rc_binomial_heap` (const Cmp_Fn &)
- `rc_binomial_heap` (const `rc_binomial_heap`< Value_Type, Cmp_Fn, _Alloc > &)
- `iterator begin` ()
- `const_iterator begin` () const
- void `clear` ()
- bool `empty` () const
- `iterator end` ()
- `const_iterator end` () const
- void `erase` (`point_iterator`)
- template<typename Pred >
size_type `erase_if` (Pred)
- Cmp_Fn & `get_cmp_fn` ()
- const Cmp_Fn & `get_cmp_fn` () const
- void `join` (`rc_binomial_heap`< Value_Type, Cmp_Fn, _Alloc > &)
- void `join` (`binomial_heap_base`< Value_Type, Cmp_Fn, _Alloc > &)
- size_type `max_size` () const
- void `modify` (`point_iterator`, const_reference)
- void `pop` ()
- `point_iterator push` (const_reference)
- size_type `size` () const
- template<typename Pred >
void `split` (Pred, `rc_binomial_heap`< Value_Type, Cmp_Fn, _Alloc > &)
- template<typename Pred >
void `split` (Pred, `binomial_heap_base`< Value_Type, Cmp_Fn, _Alloc > &)
- void `swap` (`rc_binomial_heap`< Value_Type, Cmp_Fn, _Alloc > &)
- void `swap` (`left_child_next_sibling_heap`< Value_Type, Cmp_Fn, _Alloc::size_type, _Alloc > &)
- const_reference `top` () const

Protected Types

- typedef base_type::node `node`
- typedef _Alloc::template rebind< `left_child_next_sibling_heap_node`< Value_Type, _Alloc::size_type, _Alloc >::other `node_allocator`
- typedef _Alloc::size_type `node_metadata`
- typedef `std::pair`< node_pointer, node_pointer > `node_pointer_pair`

Protected Member Functions

- void `actual_erase_node` (node_pointer)
- void `bubble_to_top` (node_pointer)
- void `clear_imp` (node_pointer)
- template<typename It >
void `copy_from_range` (It, It)
- void `find_max` ()
- node_pointer `get_new_node_for_insert` (const_reference)
- node_pointer `prune` (Pred)
- void `swap` (`binomial_heap_base`< Value_Type, Cmp_Fn, _Alloc > &)
- void `swap_with_parent` (node_pointer, node_pointer)
- void `to_linked_list` ()
- void `value_swap` (`left_child_next_sibling_heap` &)

Static Protected Member Functions

- static void **make_child_of** (node_pointer, node_pointer)
- static node_pointer **parent** (node_pointer)

Protected Attributes

- node_pointer **m_p_max**
- node_pointer **m_p_root**
- size_type **m_size**

5.316.1 Detailed Description

```
template<typename Value_Type, typename Cmp_Fn, typename _Alloc>
class __gnu_pbds::detail::rc_binomial_heap< Value_Type, Cmp_Fn, _Alloc >
```

Redundant-counter binomial heap.

Definition at line 66 of file rc_binomial_heap_.hpp.

The documentation for this class was generated from the following file:

- [rc_binomial_heap_.hpp](#)

5.317 __gnu_pbds::detail::resize_policy< _Tp > Class Template Reference

Public Types

- typedef _Tp **size_type**

Public Member Functions

- **resize_policy** (const [resize_policy](#) &other)
- size_type **get_new_size_for_arbitrary** (size_type) const
- size_type **get_new_size_for_grow** () const
- size_type **get_new_size_for_shrink** () const
- bool **grow_needed** (size_type) const
- void **notify_arbitrary** (size_type)
- void **notify_grow_resize** ()
- void **notify_shrink_resize** ()
- bool **resize_needed_for_grow** (size_type) const
- bool **resize_needed_for_shrink** (size_type) const
- bool **shrink_needed** (size_type) const
- void **swap** ([resize_policy](#)< _Tp > &)

Static Public Attributes

- static const `_Tp` **min_size**

5.317.1 Detailed Description

```
template<typename _Tp>
class __gnu_pbds::detail::resize_policy< _Tp >
```

Resize policy for binary heap.

Definition at line 52 of file `resize_policy.hpp`.

The documentation for this class was generated from the following file:

- [resize_policy.hpp](#)

5.318 `__gnu_pbds::detail::splay_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >` Class Template Reference

Inherits `__gnu_pbds::detail::bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >`.

Public Types

- typedef `_Alloc` **allocator_type**
- typedef `Cmp_Fn` **cmp_fn**
- typedef `std::pair< size_type, size_type >` **comp_hash**
- typedef `base_type::const_iterator` **const_iterator**
- typedef `base_type::const_pointer` **const_pointer**
- typedef `base_type::const_reference` **const_reference**
- typedef `base_type::const_reverse_iterator` **const_reverse_iterator**
- typedef `splay_tree_tag` **container_category**
- typedef `_Alloc::difference_type` **difference_type**
- typedef `base_type::iterator` **iterator**
- typedef `base_type::key_const_pointer` **key_const_pointer**
- typedef `base_type::key_const_reference` **key_const_reference**
- typedef `base_type::key_pointer` **key_pointer**
- typedef `base_type::key_reference` **key_reference**
- typedef `base_type::key_type` **key_type**
- typedef `base_type::mapped_const_pointer` **mapped_const_pointer**
- typedef `base_type::mapped_const_reference` **mapped_const_reference**
- typedef `base_type::mapped_pointer` **mapped_pointer**
- typedef `base_type::mapped_reference` **mapped_reference**
- typedef `base_type::mapped_type` **mapped_type**
- typedef `__nothrowcopy::indicator` **no_throw_indicator**
- typedef `traits_type::node_const_iterator` **node_const_iterator**
- typedef `traits_type::node_iterator` **node_iterator**

- typedef base_type::node_update **node_update**
- typedef base_type::const_iterator **point_const_iterator**
- typedef base_type::point_iterator **point_iterator**
- typedef base_type::pointer **pointer**
- typedef base_type::reference **reference**
- typedef base_type::reverse_iterator **reverse_iterator**
- typedef _Alloc::size_type **size_type**
- typedef integral_constant< int, Store_Hash > **store_extra**
- typedef base_type::value_type **value_type**

Public Member Functions

- **splay_tree_map** (const Cmp_Fn &)
- **splay_tree_map** (const Cmp_Fn &, const node_update &)
- **splay_tree_map** (const [splay_tree_map](#)< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)
- iterator **begin** ()
- const_iterator **begin** () const
- void **clear** ()
- template<typename It >
void **copy_from_range** (It, It)
- bool **empty** () const
- iterator **end** ()
- const_iterator **end** () const
- bool **erase** (key_const_reference)
- iterator **erase** (iterator it)
- reverse_iterator **erase** (reverse_iterator)
- template<typename Pred >
size_type **erase_if** (Pred)
- point_iterator **find** (key_const_reference)
- point_const_iterator **find** (key_const_reference) const
- Cmp_Fn & **get_cmp_fn** ()
- const Cmp_Fn & **get_cmp_fn** () const
- void **initialize** ()
- [std::pair](#)< point_iterator, bool > **insert** (const_reference r_value)
- void **join** ([splay_tree_map](#)< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)
- point_iterator **lower_bound** (key_const_reference)
- point_const_iterator **lower_bound** (key_const_reference) const
- size_type **max_size** () const
- node_const_iterator **node_begin** () const
- node_iterator **node_begin** ()
- node_const_iterator **node_end** () const
- node_iterator **node_end** ()
- mapped_reference **operator[]** (key_const_reference r_key)
- reverse_iterator **rbegin** ()
- const_reverse_iterator **rbegin** () const
- reverse_iterator **rend** ()
- const_reverse_iterator **rend** () const
- size_type **size** () const
- void **split** (key_const_reference, [splay_tree_map](#)< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)
- void **swap** ([splay_tree_map](#)< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)
- void **swap** (bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)
- point_iterator **upper_bound** (key_const_reference)
- point_const_iterator **upper_bound** (key_const_reference) const

Public Attributes

- no_throw_indicator **m_no_throw_copies_indicator**
- store_extra **m_store_extra_indicator**

Protected Types

- typedef node_allocator::value_type **node**
- typedef _Alloc::template rebind< typename traits_type::node >::other **node_allocator**
- typedef traits_type::null_node_update_pointer **null_node_update_pointer**
- typedef [types_traits](#)< Key, Mapped, _Alloc, false > **traits_base**

Protected Member Functions

- void **actual_erase_node** (node_pointer)
- void **apply_update** (node_pointer, null_node_update_pointer)
- template<typename Node_Update_>
void **apply_update** (node_pointer, Node_Update_*)
- [std::pair](#)< node_pointer, bool > **erase** (node_pointer)
- node_pointer **get_new_node_for_leaf_insert** (const_reference, false_type)
- node_pointer **get_new_node_for_leaf_insert** (const_reference, true_type)
- void **initialize_min_max** ()
- iterator **insert_imp_empty** (const_reference)
- [std::pair](#)< point_iterator, bool > **insert_leaf** (const_reference)
- iterator **insert_leaf_new** (const_reference, node_pointer, bool)
- void **join_finish** (bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)
- bool **join_prep** (bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)
- size_type **recursive_count** (node_pointer) const
- void **rotate_left** (node_pointer)
- void **rotate_parent** (node_pointer)
- void **rotate_right** (node_pointer)
- void **split_finish** (bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)
- bool **split_prep** (key_const_reference, bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)
- void **update_min_max_for_erased_node** (node_pointer)
- void **update_to_top** (node_pointer, null_node_update_pointer)
- template<typename Node_Update_>
void **update_to_top** (node_pointer, Node_Update_*)
- void **value_swap** (bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc > &)

Static Protected Member Functions

- static void **clear_imp** (node_pointer)

Protected Attributes

- node_pointer **m_p_head**
- size_type **m_size**

Static Protected Attributes

- static node_allocator **s_node_allocator**

5.318.1 Detailed Description

```
template<typename Key, typename Mapped, typename Cmp_Fn, typename Node_And_It_Traits, typename _Alloc>
class __gnu_pbds::detail::splay_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >
```

Splay tree.

Definition at line 107 of file splay_tree_.hpp.

5.318.2 Member Function Documentation

```
5.318.2.1 template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename
        _Alloc > bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_const_iterator
        __gnu_pbds::detail::bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_begin ( ) const
        [inline], [inherited]
```

Returns a const node_iterator corresponding to the node at the root of the tree.

Definition at line 109 of file bin_search_tree_.hpp.

```
5.318.2.2 template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename
        _Alloc > bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_iterator
        __gnu_pbds::detail::bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_begin ( )
        [inline], [inherited]
```

Returns a node_iterator corresponding to the node at the root of the tree.

Definition at line 117 of file bin_search_tree_.hpp.

```
5.318.2.3 template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename
        _Alloc > bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_const_iterator
        __gnu_pbds::detail::bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_end ( ) const
        [inline], [inherited]
```

Returns a const node_iterator corresponding to a node just after a leaf of the tree.

Definition at line 125 of file bin_search_tree_.hpp.

```
5.318.2.4 template<typename Key , typename Mapped , typename Cmp_Fn , typename Node_And_It_Traits , typename
    _Alloc > bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_iterator
    __gnu_pbds::detail::bin_search_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::node_end ( )
    [inline], [inherited]
```

Returns a `node_iterator` corresponding to a node just after a leaf of the tree.

Definition at line 133 of file `bin_search_tree_.hpp`.

The documentation for this class was generated from the following file:

- [splay_tree_.hpp](#)

5.319 `__gnu_pbds::detail::splay_tree_node_ < Value_Type, Metadata, _Alloc >` Struct Template Reference

Public Types

- typedef `_Alloc::template rebind< metadata_type >::other::const_reference` **metadata_const_reference**
- typedef `_Alloc::template rebind< metadata_type >::other::reference` **metadata_reference**
- typedef `Metadata` **metadata_type**
- typedef `_Alloc::template rebind< splay_tree_node_ < Value_Type, Metadata, _Alloc > >::other::pointer` **node_pointer**
- typedef `Value_Type` **value_type**

Public Member Functions

- `metadata_const_reference` **get_metadata** () const
- `metadata_reference` **get_metadata** ()
- `bool` **special** () const

Public Attributes

- `metadata_type` **m_metadata**
- `node_pointer` **m_p_left**
- `node_pointer` **m_p_parent**
- `node_pointer` **m_p_right**
- `bool` **m_special**
- `value_type` **m_value**

5.319.1 Detailed Description

```
template<typename Value_Type, class Metadata, typename _Alloc>
struct __gnu_pbds::detail::splay_tree_node_ < Value_Type, Metadata, _Alloc >
```

Node for splay tree.

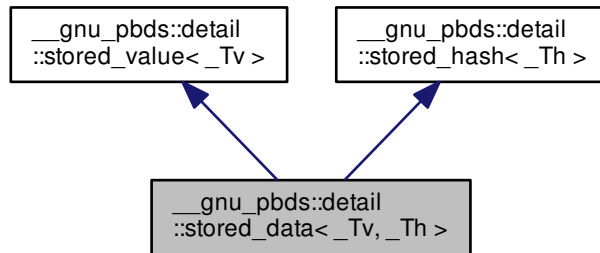
Definition at line 50 of file `splay_tree_/node.hpp`.

The documentation for this struct was generated from the following file:

- [splay_tree_/node.hpp](#)

5.320 `__gnu_pbds::detail::stored_data<_Tv,_Th>` Struct Template Reference

Inheritance diagram for `__gnu_pbds::detail::stored_data<_Tv,_Th>`:



Public Types

- typedef `_Th` **hash_type**
- typedef `_Tv` **value_type**

Public Attributes

- hash_type **m_hash**
- value_type **m_value**

5.320.1 Detailed Description

```
template<typename _Tv, typename _Th>
struct __gnu_pbds::detail::stored_data<_Tv,_Th>
```

Primary template for representation of stored data. Two types of data can be stored: value and hash.

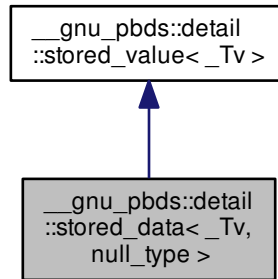
Definition at line 95 of file `types_traits.hpp`.

The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

5.321 `__gnu_pbds::detail::stored_data<_Tv, null_type>` Struct Template Reference

Inheritance diagram for `__gnu_pbds::detail::stored_data<_Tv, null_type>`:



Public Types

- typedef `_Tv` **value_type**

Public Attributes

- value_type **m_value**

5.321.1 Detailed Description

```
template<typename _Tv>
struct __gnu_pbds::detail::stored_data<_Tv, null_type>
```

Specialization for representation of stored data of just value type.

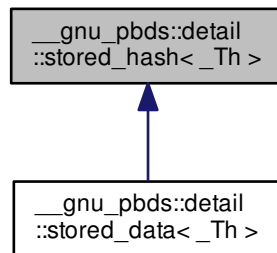
Definition at line 101 of file `types_traits.hpp`.

The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

5.322 `__gnu_pbds::detail::stored_hash<_Th>` Struct Template Reference

Inheritance diagram for `__gnu_pbds::detail::stored_hash<_Th>`:



Public Types

- typedef `_Th` **hash_type**

Public Attributes

- hash_type **m_hash**

5.322.1 Detailed Description

```
template<typename _Th>
struct __gnu_pbds::detail::stored_hash<_Th>
```

Stored hash.

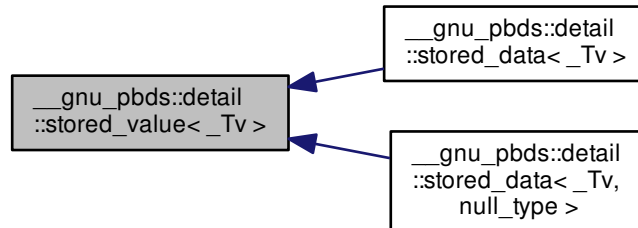
Definition at line 86 of file `types_traits.hpp`.

The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

5.323 `__gnu_pbds::detail::stored_value<_Tv>` Struct Template Reference

Inheritance diagram for `__gnu_pbds::detail::stored_value<_Tv>`:



Public Types

- typedef `_Tv` **value_type**

Public Attributes

- value_type **m_value**

5.323.1 Detailed Description

```
template<typename _Tv>
struct __gnu_pbds::detail::stored_value<_Tv>
```

Stored value.

Definition at line 78 of file `types_traits.hpp`.

The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

5.324 `__gnu_pbds::detail::synth_access_traits<Type_Traits, Set, _ATraits>` Struct Template Reference

Inherits `_ATraits`.

Public Types

- typedef `_ATraits` **base_type**
- typedef `base_type::const_iterator` **const_iterator**
- typedef `type_traits::const_reference` **const_reference**
- typedef `type_traits::key_const_reference` **key_const_reference**
- typedef `Type_Traits` **type_traits**

Public Member Functions

- **synth_access_traits** (`const base_type &`)
- **cmp_keys** (`key_const_reference, key_const_reference`) `const`
- **cmp_prefixes** (`const_iterator, const_iterator, const_iterator, const_iterator, bool compare_after=false`) `const`
- **equal_keys** (`key_const_reference, key_const_reference`) `const`
- **equal_prefixes** (`const_iterator, const_iterator, const_iterator, const_iterator, bool compare_after=true`) `const`

Static Public Member Functions

- static `key_const_reference` **extract_key** (`const_reference`)

5.324.1 Detailed Description

```
template<typename Type_Traits, bool Set, typename _ATraits>
struct __gnu_pbds::detail::synth_access_traits< Type_Traits, Set, _ATraits >
```

Synthetic element access traits.

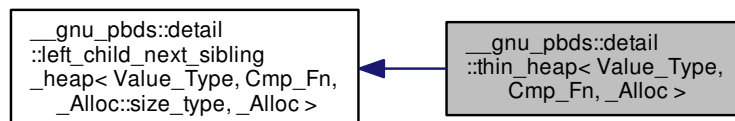
Definition at line 59 of file `synth_access_traits.hpp`.

The documentation for this struct was generated from the following file:

- [synth_access_traits.hpp](#)

5.325 `__gnu_pbds::detail::thin_heap< Value_Type, Cmp_Fn, _Alloc >` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::thin_heap< Value_Type, Cmp_Fn, _Alloc >`:



Public Types

- typedef `_Alloc` **allocator_type**
- typedef `Cmp_Fn` **cmp_fn**
- typedef `base_type::const_iterator` **const_iterator**
- typedef `__rebind_a::const_pointer` **const_pointer**
- typedef `__rebind_a::const_reference` **const_reference**
- typedef `_Alloc::difference_type` **difference_type**
- typedef `base_type::iterator` **iterator**
- typedef `base_type::point_const_iterator` **point_const_iterator**
- typedef `base_type::point_iterator` **point_iterator**
- typedef `__rebind_a::pointer` **pointer**
- typedef `__rebind_a::reference` **reference**
- typedef `_Alloc::size_type` **size_type**
- typedef `Value_Type` **value_type**

Public Member Functions

- **iterator** **begin** ()
- **const_iterator** **begin** () const
- void **clear** ()
- bool **empty** () const
- **iterator** **end** ()
- **const_iterator** **end** () const
- void **erase** (**point_iterator**)
- template<typename Pred >
size_type **erase_if** (Pred)
- `Cmp_Fn` & **get_cmp_fn** ()
- const `Cmp_Fn` & **get_cmp_fn** () const
- void **join** (**thin_heap**< `Value_Type`, `Cmp_Fn`, `_Alloc` > &)
- size_type **max_size** () const
- void **modify** (**point_iterator**, const_reference)
- void **pop** ()
- **point_iterator** **push** (const_reference)
- size_type **size** () const
- template<typename Pred >
void **split** (Pred, **thin_heap**< `Value_Type`, `Cmp_Fn`, `_Alloc` > &)
- void **swap** (**left_child_next_sibling_heap**< `Value_Type`, `Cmp_Fn`, `_Alloc::size_type`, `_Alloc` > &)
- const_reference **top** () const

Protected Types

- typedef `base_type::node` **node**
- typedef `_Alloc::template rebind< left_child_next_sibling_heap_node_< Value_Type, _Alloc::size_type, _Alloc >::other` **node_allocator**
- typedef `base_type::node_const_pointer` **node_const_pointer**
- typedef `_Alloc::size_type` **node_metadata**
- typedef `base_type::node_pointer` **node_pointer**
- typedef `std::pair< node_pointer, node_pointer >` **node_pointer_pair**

Protected Member Functions

- **thin_heap** (const Cmp_Fn &)
- **thin_heap** (const [thin_heap](#)< Value_Type, Cmp_Fn, _Alloc > &)
- void **actual_erase_node** (node_pointer)
- void **bubble_to_top** (node_pointer)
- void **clear_imp** (node_pointer)
- template<typename It >
void **copy_from_range** (It, It)
- node_pointer **get_new_node_for_insert** (const_reference)
- node_pointer **prune** (Pred)
- void **swap** ([thin_heap](#)< Value_Type, Cmp_Fn, _Alloc > &)
- void **swap_with_parent** (node_pointer, node_pointer)
- void **to_linked_list** ()
- void **value_swap** ([left_child_next_sibling_heap](#) &)

Static Protected Member Functions

- static node_pointer **parent** (node_pointer)

Protected Attributes

- node_pointer **m_p_root**
- size_type **m_size**

5.325.1 Detailed Description

```
template<typename Value_Type, typename Cmp_Fn, typename _Alloc>
class __gnu_pbds::detail::thin_heap< Value_Type, Cmp_Fn, _Alloc >
```

Thin heap.

See Tarjan and Kaplan.

Definition at line 77 of file thin_heap_.hpp.

The documentation for this class was generated from the following file:

- [thin_heap_.hpp](#)

5.326 `__gnu_pbds::detail::tree_metadata_helper< Node_Update, _BTp >` Struct Template Reference

5.326.1 Detailed Description

```
template<typename Node_Update, bool _BTp>
struct __gnu_pbds::detail::tree_metadata_helper< Node_Update, _BTp >
```

Tree metadata helper.

Definition at line 58 of file `tree_policy/node_metadata_selector.hpp`.

The documentation for this struct was generated from the following file:

- [tree_policy/node_metadata_selector.hpp](#)

5.327 `__gnu_pbds::detail::tree_metadata_helper< Node_Update, false >` Struct Template Reference

Public Types

- typedef `Node_Update::metadata_type` **type**

5.327.1 Detailed Description

```
template<typename Node_Update>
struct __gnu_pbds::detail::tree_metadata_helper< Node_Update, false >
```

Specialization, false.

Definition at line 62 of file `tree_policy/node_metadata_selector.hpp`.

The documentation for this struct was generated from the following file:

- [tree_policy/node_metadata_selector.hpp](#)

5.328 `__gnu_pbds::detail::tree_metadata_helper< Node_Update, true >` Struct Template Reference

Public Types

- typedef `null_type` **type**

5.328.1 Detailed Description

```
template<typename Node_Update>
struct __gnu_pbds::detail::tree_metadata_helper< Node_Update, true >
```

Specialization, true.

Definition at line 69 of file tree_policy/node_metadata_selector.hpp.

The documentation for this struct was generated from the following file:

- [tree_policy/node_metadata_selector.hpp](#)

5.329 __gnu_pbds::detail::tree_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc > Struct Template Reference

Public Types

- typedef [tree_metadata_helper](#)< __node_u, null_update >::type **type**

5.329.1 Detailed Description

```
template<typename Key, typename Data, typename Cmp_Fn, template< typename Node_Cltr, typename Const_Iterator, typename
Cmp_Fn_, typename _Alloc > class Node_Update, typename _Alloc>
struct __gnu_pbds::detail::tree_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc >
```

Tree node metadata dispatch.

Definition at line 84 of file tree_policy/node_metadata_selector.hpp.

The documentation for this struct was generated from the following file:

- [tree_policy/node_metadata_selector.hpp](#)

5.330 __gnu_pbds::detail::tree_traits< Key, Data, Cmp_Fn, Node_Update, Tag, _Alloc > Struct Template Reference

5.330.1 Detailed Description

```
template<typename Key, typename Data, typename Cmp_Fn, template< typename Node_Cltr, typename Node_Itr, typename Cmp_Fn_
Fn_, typename _Alloc > class Node_Update, typename Tag, typename _Alloc>
struct __gnu_pbds::detail::tree_traits< Key, Data, Cmp_Fn, Node_Update, Tag, _Alloc >
```

Tree traits class, primary template.

Definition at line 70 of file branch_policy/traits.hpp.

The documentation for this struct was generated from the following file:

- [branch_policy/traits.hpp](#)

5.331 `__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >` Struct Template Reference

Public Types

- typedef `tree_node_metadata_dispatch< Key, Mapped, Cmp_Fn, Node_Update, _Alloc >::type` **metadata_type**
- typedef `ov_tree_node_const_it_< value_type, metadata_type, _Alloc >` `node_const_iterator`
- typedef `ov_tree_node_it_< value_type, metadata_type, _Alloc >` **node_iterator**
- typedef `Node_Update< node_const_iterator, node_iterator, Cmp_Fn, _Alloc >` **node_update**
- typedef `__gnu_pbds::null_node_update< node_const_iterator, node_iterator, Cmp_Fn, _Alloc > * null_node` `←_update_pointer`

5.331.1 Detailed Description

```
template<typename Key, typename Mapped, class Cmp_Fn, template< typename Node_Cltr, class Node_Itr, class Cmp_Fn_, typename _Alloc_ > class Node_Update, typename _Alloc>
struct __gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >
```

Tree traits.

Definition at line 61 of file `ov_tree_map_/traits.hpp`.

5.331.2 Member Typedef Documentation

5.331.2.1 `template<typename Key , typename Mapped , class Cmp_Fn , template< typename Node_Cltr, class Node_Itr, class Cmp_Fn_, typename _Alloc_ > class Node_Update, typename _Alloc > typedef ov_tree_node_const_it_< value_type, metadata_type, _Alloc> __gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >::node_const_iterator`

This is an iterator to an iterator: it iterates over nodes, and de-referencing it returns one of the tree's iterators.

Definition at line 95 of file `ov_tree_map_/traits.hpp`.

The documentation for this struct was generated from the following file:

- `ov_tree_map_/traits.hpp`

5.332 `__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >` Struct Template Reference

Public Types

- typedef `tree_node_metadata_dispatch< Key, null_type, Cmp_Fn, Node_Update, _Alloc >::type` **metadata_type**
- typedef `ov_tree_node_const_it_< value_type, metadata_type, _Alloc >` `node_const_iterator`
- typedef `node_const_iterator` **node_iterator**
- typedef `Node_Update< node_const_iterator, node_const_iterator, Cmp_Fn, _Alloc >` **node_update**
- typedef `__gnu_pbds::null_node_update< node_const_iterator, node_iterator, Cmp_Fn, _Alloc > * null_node` `←_update_pointer`

5.332.1 Detailed Description

```
template<typename Key, class Cmp_Fn, template< typename Node_Cltr, class Node_Itr, class Cmp_Fn_, typename _Alloc_ > class
Node_Update, typename _Alloc>
struct __gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >
```

Specialization.

Definition at line 132 of file `ov_tree_map_/traits.hpp`.

5.332.2 Member Typedef Documentation

```
5.332.2.1 template<typename Key , class Cmp_Fn , template< typename Node_Cltr, class Node_Itr, class Cmp_Fn_, typename
_Alloc_ > class Node_Update, typename _Alloc > typedef ov_tree_node_const_it_< value_type, metadata_type,
_Alloc> __gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc
>::node_const_iterator
```

This is an iterator to an iterator: it iterates over nodes, and de-referencing it returns one of the tree's iterators.

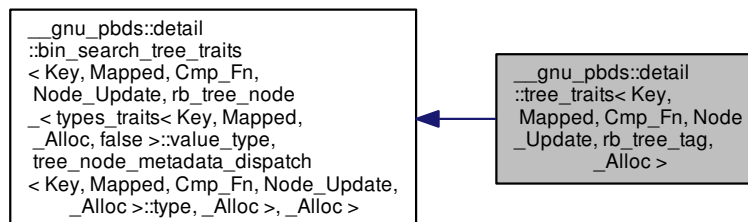
Definition at line 166 of file `ov_tree_map_/traits.hpp`.

The documentation for this struct was generated from the following file:

- [ov_tree_map_/traits.hpp](#)

5.333 __gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc > Struct Template Reference

Inheritance diagram for `__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >`:



Public Types

- typedef `bin_search_tree_const_it_< typename _Alloc::template rebind< node >::other::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, false, _Alloc >` **const_reverse_iterator**
- typedef `rb_tree_node_< types_traits< Key, Mapped, _Alloc, false >::value_type, tree_node_metadata_dispatch< Key, Mapped, Cmp_Fn, Node_Update, _Alloc >::type, _Alloc >` **node**
- typedef `bin_search_tree_const_node_it_< rb_tree_node_< types_traits< Key, Mapped, _Alloc, false >::value_type, tree_node_metadata_dispatch< Key, Mapped, Cmp_Fn, Node_Update, _Alloc >::type, _Alloc >` **point_const_iterator**, `point_iterator`, `_Alloc >` **node_const_iterator**
- typedef `bin_search_tree_node_it_< rb_tree_node_< types_traits< Key, Mapped, _Alloc, false >::value_type, tree_node_metadata_dispatch< Key, Mapped, Cmp_Fn, Node_Update, _Alloc >::type, _Alloc >` **point_const_iterator**, `point_iterator`, `_Alloc >` **node_iterator**
- typedef `Node_Update< node_const_iterator, node_iterator, Cmp_Fn, _Alloc >` **node_update**
- typedef `__gnu_pbds::null_node_update< node_const_iterator, node_iterator, Cmp_Fn, _Alloc > * null_node_update_pointer`
- typedef `bin_search_tree_const_it_< typename _Alloc::template rebind< node >::other::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, true, _Alloc >` **point_const_iterator**
- typedef `bin_search_tree_it_< typename _Alloc::template rebind< node >::other::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, true, _Alloc >` **point_iterator**
- typedef `bin_search_tree_it_< typename _Alloc::template rebind< node >::other::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, false, _Alloc >` **reverse_iterator**

5.333.1 Detailed Description

```
template<typename Key, typename Mapped, typename Cmp_Fn, template< typename Node_Cltr, typename Node_Itr, typename Cmp_Fn, typename _Alloc > class Node_Update, typename _Alloc>
```

```
struct __gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >
```

Specialization.

Definition at line 61 of file `rb_tree_map_/traits.hpp`.

5.333.2 Member Typedef Documentation

- 5.333.2.1 `typedef bin_search_tree_const_node_it_< rb_tree_node_< types_traits< Key, Mapped, _Alloc, false >::value_type, tree_node_metadata_dispatch< Key, Mapped, Cmp_Fn, Node_Update, _Alloc >::type, _Alloc >` **point_const_iterator**, `point_iterator`, `_Alloc >` `__gnu_pbds::detail::bin_search_tree_traits< Key, Mapped, Cmp_Fn, Node_Update, rb_tree_node_< types_traits< Key, Mapped, _Alloc, false >::value_type, tree_node_metadata_dispatch< Key, Mapped, Cmp_Fn, Node_Update, _Alloc >::type, _Alloc >` **node_const_iterator** `[inherited]`

This is an iterator to an iterator: it iterates over nodes, and de-referencing it returns one of the tree's iterators.

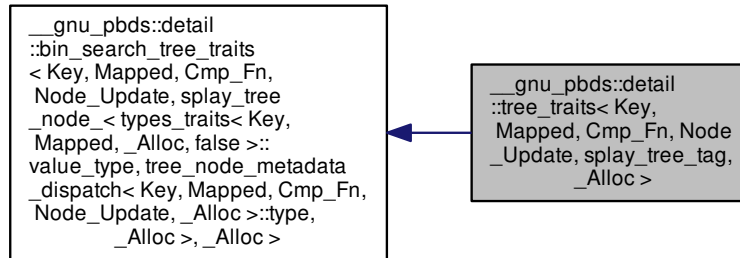
Definition at line 131 of file `bin_search_tree_/traits.hpp`.

The documentation for this struct was generated from the following file:

- [rb_tree_map_/traits.hpp](#)

5.334 `__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >` Struct Template Reference

Inheritance diagram for `__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >`:



Public Types

- typedef `bin_search_tree_const_it` < typename `_Alloc::template rebind< node >::other::pointer`, typename `type_traits::value_type`, typename `type_traits::pointer`, typename `type_traits::const_pointer`, typename `type_traits::reference`, typename `type_traits::const_reference`, false, `_Alloc` > **const_reverse_iterator**
- typedef `splay_tree_node` < `types_traits< Key, Mapped, _Alloc, false >::value_type`, `tree_node_metadata_dispatch` < `Key, Mapped, Cmp_Fn, Node_Update, _Alloc >::type`, `_Alloc` > **node**
- typedef `bin_search_tree_const_node_it` < `splay_tree_node` < `types_traits< Key, Mapped, _Alloc, false >::value_type`, `tree_node_metadata_dispatch` < `Key, Mapped, Cmp_Fn, Node_Update, _Alloc >::type`, `_Alloc` >, `point_const_iterator`, `point_iterator`, `_Alloc` > **node_const_iterator**
- typedef `bin_search_tree_node_it` < `splay_tree_node` < `types_traits< Key, Mapped, _Alloc, false >::value_type`, `tree_node_metadata_dispatch` < `Key, Mapped, Cmp_Fn, Node_Update, _Alloc >::type`, `_Alloc` >, `point_const_iterator`, `point_iterator`, `_Alloc` > **node_iterator**
- typedef `Node_Update` < `node_const_iterator`, `node_iterator`, `Cmp_Fn`, `_Alloc` > **node_update**
- typedef `__gnu_pbds::null_node_update` < `node_const_iterator`, `node_iterator`, `Cmp_Fn`, `_Alloc` > * **null_node_update_pointer**
- typedef `bin_search_tree_const_it` < typename `_Alloc::template rebind< node >::other::pointer`, typename `type_traits::value_type`, typename `type_traits::pointer`, typename `type_traits::const_pointer`, typename `type_traits::reference`, typename `type_traits::const_reference`, true, `_Alloc` > **point_const_iterator**
- typedef `bin_search_tree_it` < typename `_Alloc::template rebind< node >::other::pointer`, typename `type_traits::value_type`, typename `type_traits::pointer`, typename `type_traits::const_pointer`, typename `type_traits::reference`, typename `type_traits::const_reference`, true, `_Alloc` > **point_iterator**
- typedef `bin_search_tree_it` < typename `_Alloc::template rebind< node >::other::pointer`, typename `type_traits::value_type`, typename `type_traits::pointer`, typename `type_traits::const_pointer`, typename `type_traits::reference`, typename `type_traits::const_reference`, false, `_Alloc` > **reverse_iterator**

5.334.1 Detailed Description

```
template<typename Key, typename Mapped, typename Cmp_Fn, template< typename Node_Cltr, typename Node_Itr, typename
Cmp_Fn_, typename _Alloc > class Node_Update, typename _Alloc>
struct __gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >
```

Specialization.

Definition at line 61 of file `splay_tree_/traits.hpp`.

5.334.2 Member Typedef Documentation

5.334.2.1 `typedef bin_search_tree_const_node_it_< splay_tree_node_< types_traits< Key, Mapped, _Alloc, false >::value_type, tree_node_metadata_dispatch< Key, Mapped, Cmp_Fn, Node_Update, _Alloc >::type, _Alloc >, point_const_iterator, point_iterator, _Alloc> __gnu_pbds::detail::bin_search_tree_traits< Key, Mapped, Cmp_Fn, Node_Update, splay_tree_node_< types_traits< Key, Mapped, _Alloc, false >::value_type, tree_node_metadata_dispatch< Key, Mapped, Cmp_Fn, Node_Update, _Alloc >::type, _Alloc >, _Alloc >::node_const_iterator` [inherited]

This is an iterator to an iterator: it iterates over nodes, and de-referencing it returns one of the tree's iterators.

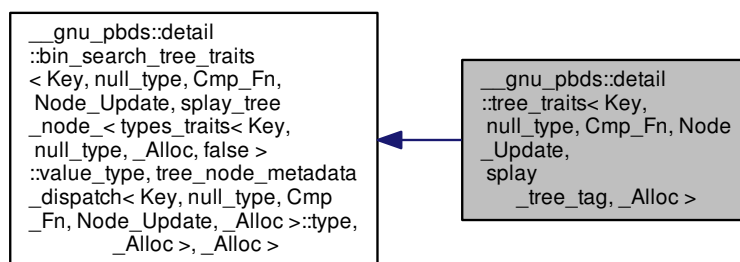
Definition at line 131 of file `bin_search_tree_/traits.hpp`.

The documentation for this struct was generated from the following file:

- [splay_tree_/traits.hpp](#)

5.335 `__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >` Struct Template Reference

Inheritance diagram for `__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >`:



Public Types

- typedef [bin_search_tree_const_it](#) < typename [_Alloc](#)::template rebind< [node](#) >::other::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, false, [_Alloc](#) > **const_reverse_iterator**
- typedef [splay_tree_node](#) < [types_traits](#)< Key, [null_type](#), [_Alloc](#), false >::value_type, [tree_node_metadata_dispatch](#) < Key, [null_type](#), [Cmp_Fn](#), [Node_Update](#), [_Alloc](#) >::type, [_Alloc](#) > **node**
- typedef [bin_search_tree_const_node_it](#) < [splay_tree_node](#) < [types_traits](#)< Key, [null_type](#), [_Alloc](#), false >::value_type, [tree_node_metadata_dispatch](#) < Key, [null_type](#), [Cmp_Fn](#), [Node_Update](#), [_Alloc](#) >::type, [_Alloc](#) >, [point_const_iterator](#), [point_iterator](#), [_Alloc](#) > **node_const_iterator**
- typedef [bin_search_tree_node_it](#) < [splay_tree_node](#) < [types_traits](#)< Key, [null_type](#), [_Alloc](#), false >::value_type, [tree_node_metadata_dispatch](#) < Key, [null_type](#), [Cmp_Fn](#), [Node_Update](#), [_Alloc](#) >::type, [_Alloc](#) >, [point_const_iterator](#), [point_iterator](#), [_Alloc](#) > **node_iterator**
- typedef [Node_Update](#) < [node_const_iterator](#), [node_iterator](#), [Cmp_Fn](#), [_Alloc](#) > **node_update**
- typedef [__gnu_pbds::null_node_update](#) < [node_const_iterator](#), [node_iterator](#), [Cmp_Fn](#), [_Alloc](#) > * **null_node_update_pointer**
- typedef [bin_search_tree_const_it](#) < typename [_Alloc](#)::template rebind< [node](#) >::other::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, true, [_Alloc](#) > **point_const_iterator**
- typedef [bin_search_tree_it](#) < typename [_Alloc](#)::template rebind< [node](#) >::other::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, true, [_Alloc](#) > **point_iterator**
- typedef [bin_search_tree_it](#) < typename [_Alloc](#)::template rebind< [node](#) >::other::pointer, typename type_traits::value_type, typename type_traits::pointer, typename type_traits::const_pointer, typename type_traits::reference, typename type_traits::const_reference, false, [_Alloc](#) > **reverse_iterator**

5.335.1 Detailed Description

```
template<typename Key, class Cmp_Fn, template< typename Node_Cltr, class Node_Itr, class Cmp_Fn_, typename _Alloc > class Node_Update, typename _Alloc>
```

```
struct __gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >
```

Specialization.

Definition at line 81 of file [splay_tree_/traits.hpp](#).

5.335.2 Member Typedef Documentation

- 5.335.2.1 typedef [bin_search_tree_const_node_it](#) < [splay_tree_node](#) < [types_traits](#)< Key, [null_type](#), [_Alloc](#), false >::value_type, [tree_node_metadata_dispatch](#) < Key, [null_type](#), [Cmp_Fn](#), [Node_Update](#), [_Alloc](#) >::type, [_Alloc](#) >, [point_const_iterator](#), [point_iterator](#), [_Alloc](#) > [__gnu_pbds::detail::bin_search_tree_traits](#) < Key, [null_type](#), [Cmp_Fn](#), [Node_Update](#), [splay_tree_node](#) < [types_traits](#)< Key, [null_type](#), [_Alloc](#), false >::value_type, [tree_node_metadata_dispatch](#) < Key, [null_type](#), [Cmp_Fn](#), [Node_Update](#), [_Alloc](#) >::type, [_Alloc](#) >, [_Alloc](#) >::**node_const_iterator** [\[inherited\]](#)

This is an iterator to an iterator: it iterates over nodes, and de-referencing it returns one of the tree's iterators.

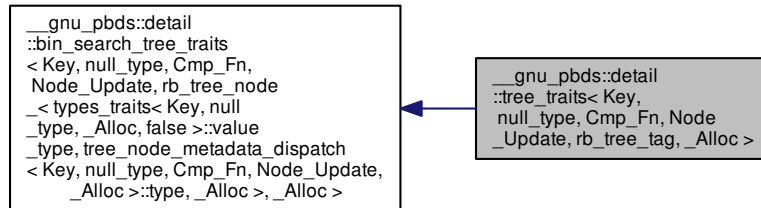
Definition at line 131 of file [bin_search_tree_/traits.hpp](#).

The documentation for this struct was generated from the following file:

- [splay_tree_/traits.hpp](#)

5.336 `__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >` Struct Template Reference

Inheritance diagram for `__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >`:



Public Types

- typedef `bin_search_tree_const_it` < typename `_Alloc::template rebind< node >::other::pointer`, typename `type_traits::value_type`, typename `type_traits::pointer`, typename `type_traits::const_pointer`, typename `type_traits::reference`, typename `type_traits::const_reference`, false, `_Alloc` > **const_reverse_iterator**
- typedef `rb_tree_node` < `types_traits< Key, null_type, _Alloc, false >::value_type`, `tree_node_metadata_dispatch< Key, null_type, Cmp_Fn, Node_Update, _Alloc >::type`, `_Alloc` > **node**
- typedef `bin_search_tree_const_node_it` < `rb_tree_node` < `types_traits< Key, null_type, _Alloc, false >::value_type`, `tree_node_metadata_dispatch< Key, null_type, Cmp_Fn, Node_Update, _Alloc >::type`, `_Alloc` >, `point_const_iterator`, `point_iterator`, `_Alloc` > **node_const_iterator**
- typedef `bin_search_tree_node_it` < `rb_tree_node` < `types_traits< Key, null_type, _Alloc, false >::value_type`, `tree_node_metadata_dispatch< Key, null_type, Cmp_Fn, Node_Update, _Alloc >::type`, `_Alloc` >, `point_const_iterator`, `point_iterator`, `_Alloc` > **node_iterator**
- typedef `Node_Update` < `node_const_iterator`, `node_iterator`, `Cmp_Fn`, `_Alloc` > **node_update**
- typedef `__gnu_pbds::null_node_update` < `node_const_iterator`, `node_iterator`, `Cmp_Fn`, `_Alloc` > * **null_node_update_pointer**
- typedef `bin_search_tree_const_it` < typename `_Alloc::template rebind< node >::other::pointer`, typename `type_traits::value_type`, typename `type_traits::pointer`, typename `type_traits::const_pointer`, typename `type_traits::reference`, typename `type_traits::const_reference`, true, `_Alloc` > **point_const_iterator**
- typedef `bin_search_tree_it` < typename `_Alloc::template rebind< node >::other::pointer`, typename `type_traits::value_type`, typename `type_traits::pointer`, typename `type_traits::const_pointer`, typename `type_traits::reference`, typename `type_traits::const_reference`, true, `_Alloc` > **point_iterator**
- typedef `bin_search_tree_it` < typename `_Alloc::template rebind< node >::other::pointer`, typename `type_traits::value_type`, typename `type_traits::pointer`, typename `type_traits::const_pointer`, typename `type_traits::reference`, typename `type_traits::const_reference`, false, `_Alloc` > **reverse_iterator**

5.336.1 Detailed Description

```

template<typename Key, typename Cmp_Fn, template< typename Node_Cltr, typename Node_Itr, typename Cmp_Fn, typename _Alloc > class Node_Update, typename _Alloc>
struct __gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >

```

Specialization.

Definition at line 85 of file `rb_tree_map_/traits.hpp`.

5.336.2 Member Typedef Documentation

5.336.2.1 `typedef bin_search_tree_const_node_it_< rb_tree_node_< types_traits< Key, null_type, _Alloc, false >::value_type, tree_node_metadata_dispatch< Key, null_type, Cmp_Fn, Node_Update, _Alloc >::type, _Alloc >, point_const_iterator, point_iterator, _Alloc> __gnu_pbds::detail::bin_search_tree_traits< Key, null_type, Cmp_Fn, Node_Update, rb_tree_node_< types_traits< Key, null_type, _Alloc, false >::value_type, tree_node_metadata_dispatch< Key, null_type, Cmp_Fn, Node_Update, _Alloc >::type, _Alloc >, _Alloc >::node_const_iterator` [inherited]

This is an iterator to an iterator: it iterates over nodes, and de-referencing it returns one of the tree's iterators.

Definition at line 131 of file `bin_search_tree_/traits.hpp`.

The documentation for this struct was generated from the following file:

- [rb_tree_map_/traits.hpp](#)

5.337 __gnu_pbds::detail::trie_metadata_helper< Node_Update, _BTp > Struct Template Reference

5.337.1 Detailed Description

```
template<typename Node_Update, bool _BTp>
struct __gnu_pbds::detail::trie_metadata_helper< Node_Update, _BTp >
```

Trie metadata helper.

Definition at line 58 of file `trie_policy/node_metadata_selector.hpp`.

The documentation for this struct was generated from the following file:

- [trie_policy/node_metadata_selector.hpp](#)

5.338 __gnu_pbds::detail::trie_metadata_helper< Node_Update, false > Struct Template Reference

Public Types

- `typedef Node_Update::metadata_type type`

5.338.1 Detailed Description

```
template<typename Node_Update>
struct __gnu_pbds::detail::trie_metadata_helper< Node_Update, false >
```

Specialization, false.

Definition at line 62 of file `trie_policy/node_metadata_selector.hpp`.

The documentation for this struct was generated from the following file:

- [trie_policy/node_metadata_selector.hpp](#)

5.339 `__gnu_pbds::detail::trie_metadata_helper< Node_Update, true >` Struct Template Reference

Public Types

- typedef `null_type` **type**

5.339.1 Detailed Description

```
template<typename Node_Update>
struct __gnu_pbds::detail::trie_metadata_helper< Node_Update, true >
```

Specialization, true.

Definition at line 69 of file `trie_policy/node_metadata_selector.hpp`.

The documentation for this struct was generated from the following file:

- [trie_policy/node_metadata_selector.hpp](#)

5.340 `__gnu_pbds::detail::trie_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc >` Struct Template Reference

Public Types

- typedef `trie_metadata_helper< __node_u, null_update >::type` **type**

5.340.1 Detailed Description

```
template<typename Key, typename Data, typename Cmp_Fn, template< typename Node_Cltr, typename Const_Iterator, typename
Cmp_Fn_, typename _Alloc_ > class Node_Update, typename _Alloc>
struct __gnu_pbds::detail::trie_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc >
```

Trie node metadata dispatch.

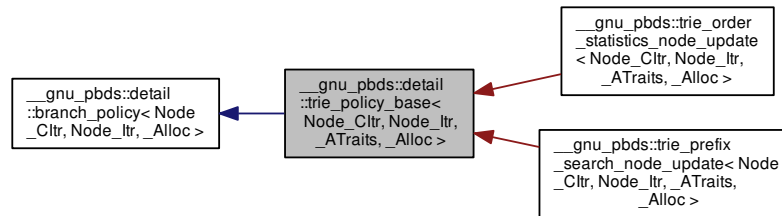
Definition at line 84 of file `trie_policy/node_metadata_selector.hpp`.

The documentation for this struct was generated from the following file:

- [trie_policy/node_metadata_selector.hpp](#)

5.341 `__gnu_pbds::detail::trie_policy_base< Node_Cltr, Node_Itr, _ATraits, _Alloc >` Class Template Reference

Inheritance diagram for `__gnu_pbds::detail::trie_policy_base< Node_Cltr, Node_Itr, _ATraits, _Alloc >`:



Public Types

- typedef `_ATraits` **access_traits**
- typedef `_Alloc` **allocator_type**
- typedef `node_const_iterator::value_type` **const_iterator**
- typedef `node_iterator::value_type` **iterator**
- typedef `base_type::key_const_reference` **key_const_reference**
- typedef `base_type::key_type` **key_type**
- typedef `null_type` **metadata_type**
- typedef `Node_Cltr` **node_const_iterator**
- typedef `Node_Itr` **node_iterator**
- typedef `allocator_type::size_type` **size_type**

Protected Types

- typedef `rebind_v::const_pointer` **const_pointer**
- typedef `rebind_v::const_reference` **const_reference**
- typedef `Node_Itr::value_type` **it_type**
- typedef `remove_const< key_type >::type` **rckey_type**
- typedef `remove_const< value_type >::type` **rcvalue_type**
- typedef `_Alloc::template rebind< rckey_type >::other` **rebind_k**
- typedef `_Alloc::template rebind< rcvalue_type >::other` **rebind_v**
- typedef `rebind_v::reference` **reference**
- typedef `std::iterator_traits< it_type >::value_type` **value_type**

Protected Member Functions

- virtual `const_iterator` **end** () const =0
- virtual `iterator` **end** ()=0
- `it_type` **end_iterator** () const
- virtual `const access_traits & get_access_traits` () const =0
- virtual `node_const_iterator` **node_begin** () const =0
- virtual `node_iterator` **node_begin** ()=0
- virtual `node_const_iterator` **node_end** () const =0
- virtual `node_iterator` **node_end** ()=0

Static Protected Member Functions

- static `size_type` **common_prefix_len** (`node_iterator`, `e_const_iterator`, `e_const_iterator`, `const access_traits &`)
- static `key_const_reference` **extract_key** (`const_reference r_val`)
- static `iterator` **leftmost_it** (`node_iterator`)
- static `bool` **less** (`e_const_iterator`, `e_const_iterator`, `e_const_iterator`, `e_const_iterator`, `const access_traits &`)
- static `iterator` **rightmost_it** (`node_iterator`)

5.341.1 Detailed Description

```
template<typename Node_Cltr, typename Node_Itr, typename _ATraits, typename _Alloc>
class __gnu_pbds::detail::trie_policy_base< Node_Cltr, Node_Itr, _ATraits, _Alloc >
```

Base class for trie policies.

Definition at line 53 of file `trie_policy_base.hpp`.

The documentation for this class was generated from the following file:

- [trie_policy_base.hpp](#)

5.342 `__gnu_pbds::detail::trie_traits< Key, Data, _ATraits, Node_Update, Tag, _Alloc >` Struct Template Reference

5.342.1 Detailed Description

```
template<typename Key, typename Data, typename _ATraits, template< typename Node_Cltr, typename Node_Itr, typename _A↵
Traits_, typename _Alloc > class Node_Update, typename Tag, typename _Alloc>
struct __gnu_pbds::detail::trie_traits< Key, Data, _ATraits, Node_Update, Tag, _Alloc >
```

Trie traits class, primary template.

Definition at line 83 of file `branch_policy/traits.hpp`.

The documentation for this struct was generated from the following file:

- [branch_policy/traits.hpp](#)

5.343 `__gnu_pbds::detail::trie_traits< Key, Mapped, _ATraits, Node_Update, pat_trie_tag, _Alloc >` Struct Template Reference

Public Types

- typedef `_ATraits` **access_traits**
- typedef `base_type::Cltr< node, leaf, head, inode, true >` **const_iterator**
- typedef `base_type::Cltr< node, leaf, head, inode, false >` **const_reverse_iterator**
- typedef `base_type::Head< synth_access_traits, metadata >` **head**
- typedef `base_type::Inode< synth_access_traits, metadata >` **inode**
- typedef `base_type::Iter< node, leaf, head, inode, true >` **iterator**
- typedef `base_type::Leaf< synth_access_traits, metadata >` **leaf**
- typedef `base_type::Metadata< metadata_type, _Alloc >` **metadata**
- typedef `trie_node_metadata_dispatch< Key, Mapped, _ATraits, Node_Update, _Alloc >::type` **metadata_type**
- typedef `base_type::Node_base< synth_access_traits, metadata >` **node**
- typedef `base_type::Node_citer< node, leaf, head, inode, const_iterator, iterator, _Alloc >` **node_const_iterator**
- typedef `base_type::Node_iter< node, leaf, head, inode, const_iterator, iterator, _Alloc >` **node_iterator**
- typedef `Node_Update< node_const_iterator, node_iterator, _ATraits, _Alloc >` **node_update**
- typedef `null_node_update< node_const_iterator, node_iterator, _ATraits, _Alloc > *` **null_node_update_pointer**
- typedef `base_type::Iter< node, leaf, head, inode, false >` **reverse_iterator**
- typedef `__gnu_pbds::detail::synth_access_traits< type_traits, false, access_traits >` **synth_access_traits**

5.343.1 Detailed Description

```
template<typename Key, typename Mapped, typename _ATraits, template< typename Node_Cltr, typename Node_Itr, typename Cmp_Fn_,
typename _Alloc_ > class Node_Update, typename _Alloc_>
struct __gnu_pbds::detail::trie_traits< Key, Mapped, _ATraits, Node_Update, pat_trie_tag, _Alloc >
```

Specialization.

Definition at line 62 of file `pat_trie_/traits.hpp`.

5.343.2 Member Typedef Documentation

5.343.2.1 `template<typename Key , typename Mapped , typename _ATraits , template< typename Node_Cltr, typename Node_Itr, typename Cmp_Fn_, typename _Alloc_ > class Node_Update, typename _Alloc_ > typedef base_type::Node_citer<node, leaf, head, inode, const_iterator, iterator, _Alloc> __gnu_pbds::detail::trie_traits< Key, Mapped, _ATraits, Node_Update, pat_trie_tag, _Alloc>::node_const_iterator`

This is an iterator to an iterator: it iterates over nodes, and de-referencing it returns one of the tree's iterators.

Definition at line 88 of file `pat_trie_/traits.hpp`.

5.343.2.2 `template<typename Key , typename Mapped , typename _ATraits , template< typename Node_Cltr, typename Node_Itr, typename Cmp_Fn_, typename _Alloc_ > class Node_Update, typename _Alloc > typedef Node_Update<node_const_iterator, node_iterator, _ATraits, _Alloc> __gnu_pbds::detail::trie_traits< Key, Mapped, _ATraits, Node_Update, pat_trie_tag, _Alloc >::node_update`

Type for node update.

Definition at line 93 of file `pat_trie_/traits.hpp`.

5.343.2.3 `template<typename Key , typename Mapped , typename _ATraits , template< typename Node_Cltr, typename Node_Itr, typename Cmp_Fn_, typename _Alloc_ > class Node_Update, typename _Alloc > typedef __gnu_pbds::detail::synth_access_traits<type_traits, false, access_traits> __gnu_pbds::detail::trie_traits< Key, Mapped, _ATraits, Node_Update, pat_trie_tag, _Alloc >::synth_access_traits`

Type for synthesized traits.

Definition at line 74 of file `pat_trie_/traits.hpp`.

The documentation for this struct was generated from the following file:

- [pat_trie_/traits.hpp](#)

5.344 `__gnu_pbds::detail::trie_traits< Key, null_type, _ATraits, Node_Update, pat_trie_tag, _Alloc >` Struct Template Reference

Public Types

- typedef `_ATraits` **access_traits**
- typedef `base_type::_Cltr< node, leaf, head, inode, true >` **const_iterator**
- typedef `base_type::_Cltr< node, leaf, head, inode, false >` **const_reverse_iterator**
- typedef `base_type::_Head< synth_access_traits, metadata >` **head**
- typedef `base_type::_Inode< synth_access_traits, metadata >` **inode**
- typedef `const_iterator` **iterator**
- typedef `base_type::_Leaf< synth_access_traits, metadata >` **leaf**
- typedef `base_type::_Metadata< metadata_type, _Alloc >` **metadata**
- typedef `trie_node_metadata_dispatch< Key, null_type, _ATraits, Node_Update, _Alloc >::type` **metadata_type**
- typedef `base_type::_Node_base< synth_access_traits, metadata >` **node**
- typedef `base_type::_Node_citer< node, leaf, head, inode, const_iterator, iterator, _Alloc >` **node_const_iterator**
- typedef `node_const_iterator` **node_iterator**
- typedef `Node_Update< node_const_iterator, node_iterator, _ATraits, _Alloc >` **node_update**
- typedef `null_node_update< node_const_iterator, node_const_iterator, _ATraits, _Alloc > * null_node_update` **←_pointer**
- typedef `const_reverse_iterator` **reverse_iterator**
- typedef `__gnu_pbds::detail::synth_access_traits< type_traits, true, access_traits >` **synth_access_traits**

5.344.1 Detailed Description

```
template<typename Key, typename _ATraits, template< typename Node_Cltr, typename Node_Itr, typename Cmp_Fn_, typename _↵
_Alloc_ > class Node_Update, typename _Alloc>
struct __gnu_pbds::detail::trie_traits< Key, null_type, _ATraits, Node_Update, pat_trie_tag, _Alloc >
```

Specialization.

Definition at line 109 of file pat_trie_/traits.hpp.

5.344.2 Member Typedef Documentation

```
5.344.2.1 template<typename Key , typename _ATraits , template< typename Node_Cltr, typename Node_Itr, typename Cmp_Fn_,
typename _Alloc_ > class Node_Update, typename _Alloc > typedef base_type::_Node_citer<node, leaf,
head, inode, const_iterator, iterator, _Alloc> __gnu_pbds::detail::trie_traits< Key, null_type, _ATraits,
Node_Update, pat_trie_tag, _Alloc >::node_const_iterator
```

This is an iterator to an iterator: it iterates over nodes, and de-referencing it returns one of the tree's iterators.

Definition at line 135 of file pat_trie_/traits.hpp.

```
5.344.2.2 template<typename Key , typename _ATraits , template< typename Node_Cltr, typename Node_Itr, typename Cmp_Fn_,
typename _Alloc_ > class Node_Update, typename _Alloc > typedef Node_Update<node_const_iterator,
node_iterator, _ATraits, _Alloc> __gnu_pbds::detail::trie_traits< Key, null_type, _ATraits, Node_Update,
pat_trie_tag, _Alloc >::node_update
```

Type for node update.

Definition at line 140 of file pat_trie_/traits.hpp.

```
5.344.2.3 template<typename Key , typename _ATraits , template< typename Node_Cltr, typename Node_Itr, typename Cmp_Fn_,
typename _Alloc_ > class Node_Update, typename _Alloc > typedef __gnu_pbds::detail::synth_access↵
_traits<type_traits, true, access_traits> __gnu_pbds::detail::trie_traits< Key, null_type, _ATraits,
Node_Update, pat_trie_tag, _Alloc >::synth_access_traits
```

Type for synthesized traits.

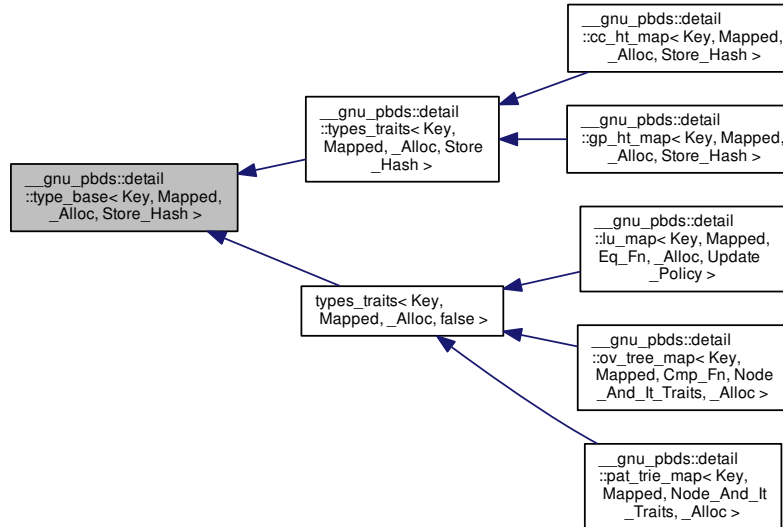
Definition at line 121 of file pat_trie_/traits.hpp.

The documentation for this struct was generated from the following file:

- [pat_trie_/traits.hpp](#)

5.345 `__gnu_pbds::detail::type_base< Key, Mapped, _Alloc, Store_Hash >` Struct Template Reference

Inheritance diagram for `__gnu_pbds::detail::type_base< Key, Mapped, _Alloc, Store_Hash >`:



5.345.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc, bool Store_Hash>
struct __gnu_pbds::detail::type_base< Key, Mapped, _Alloc, Store_Hash >
```

Primary template.

Definition at line 107 of file `types_traits.hpp`.

The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

5.346 `__gnu_pbds::detail::type_base< Key, Mapped, _Alloc, false >` Struct Template Reference

Public Types

- typedef `__rebind_va::const_pointer` **const_pointer**
- typedef `__rebind_va::const_reference` **const_reference**
- typedef `__rebind_ma::const_pointer` **mapped_const_pointer**
- typedef `__rebind_ma::const_reference` **mapped_const_reference**
- typedef `__rebind_ma::pointer` **mapped_pointer**

- typedef __rebind_ma::reference **mapped_reference**
- typedef __rebind_ma::value_type **mapped_type**
- typedef __rebind_va::pointer **pointer**
- typedef __rebind_va::reference **reference**
- typedef _Alloc::size_type **size_type**
- typedef [stored_data](#)< value_type, [null_type](#) > **stored_data_type**
- typedef __rebind_va::value_type **value_type**

5.346.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc>
struct __gnu_pbds::detail::type_base< Key, Mapped, _Alloc, false >
```

Specialization of type_base for the case where the hash value is not stored alongside each value.

Definition at line 114 of file types_traits.hpp.

The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

5.347 __gnu_pbds::detail::type_base< Key, Mapped, _Alloc, true > Struct Template Reference

Public Types

- typedef __rebind_va::const_pointer **const_pointer**
- typedef __rebind_va::const_reference **const_reference**
- typedef __rebind_ma::const_pointer **mapped_const_pointer**
- typedef __rebind_ma::const_reference **mapped_const_reference**
- typedef __rebind_ma::pointer **mapped_pointer**
- typedef __rebind_ma::reference **mapped_reference**
- typedef __rebind_ma::value_type **mapped_type**
- typedef __rebind_va::pointer **pointer**
- typedef __rebind_va::reference **reference**
- typedef _Alloc::size_type **size_type**
- typedef [stored_data](#)< value_type, size_type > **stored_data_type**
- typedef __rebind_va::value_type **value_type**

5.347.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc>
struct __gnu_pbds::detail::type_base< Key, Mapped, _Alloc, true >
```

Specialization of type_base for the case where the hash value is stored alongside each value.

Definition at line 147 of file types_traits.hpp.

The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

5.348 `__gnu_pbds::detail::type_base< Key, null_type, _Alloc, false >` Struct Template Reference

Public Types

- `typedef __rebind_va::const_pointer const_pointer`
- `typedef __rebind_va::const_reference const_reference`
- `typedef __rebind_ma::const_pointer mapped_const_pointer`
- `typedef __rebind_ma::const_reference mapped_const_reference`
- `typedef __rebind_ma::pointer mapped_pointer`
- `typedef __rebind_ma::reference mapped_reference`
- `typedef __rebind_ma::value_type mapped_type`
- `typedef __rebind_va::pointer pointer`
- `typedef __rebind_va::reference reference`
- `typedef _Alloc::size_type size_type`
- `typedef stored_data< value_type, null_type > stored_data_type`
- `typedef Key value_type`

Static Public Attributes

- static [null_type](#) **s_null_type**

5.348.1 Detailed Description

```
template<typename Key, typename _Alloc>
struct __gnu_pbds::detail::type_base< Key, null_type, _Alloc, false >
```

Specialization of `type_base` for the case where the hash value is not stored alongside each value.

Definition at line 181 of file `types_traits.hpp`.

The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

5.349 `__gnu_pbds::detail::type_base< Key, null_type, _Alloc, true >` Struct Template Reference

Public Types

- `typedef __rebind_va::const_pointer const_pointer`
- `typedef __rebind_va::const_reference const_reference`
- `typedef __rebind_ma::const_pointer mapped_const_pointer`
- `typedef __rebind_ma::const_reference mapped_const_reference`
- `typedef __rebind_ma::pointer mapped_pointer`
- `typedef __rebind_ma::reference mapped_reference`
- `typedef __rebind_ma::value_type mapped_type`
- `typedef __rebind_va::pointer pointer`
- `typedef __rebind_va::reference reference`
- `typedef _Alloc::size_type size_type`
- `typedef stored_data< value_type, size_type > stored_data_type`
- `typedef Key value_type`

Static Public Attributes

- static [null_type](#) **s_null_type**

5.349.1 Detailed Description

```
template<typename Key, typename _Alloc>
struct __gnu_pbds::detail::type_base< Key, null_type, _Alloc, true >
```

Specialization of type_base for the case where the hash value is stored alongside each value.

Definition at line 220 of file types_traits.hpp.

The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

5.350 __gnu_pbds::detail::type_dispatch< Key, Mapped, _Alloc, Store_Hash > Struct Template Reference

Public Types

- typedef [type_base](#)< Key, Mapped, _Alloc, Store_Hash > **type**

5.350.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc, bool Store_Hash>
struct __gnu_pbds::detail::type_dispatch< Key, Mapped, _Alloc, Store_Hash >
```

Type base dispatch.

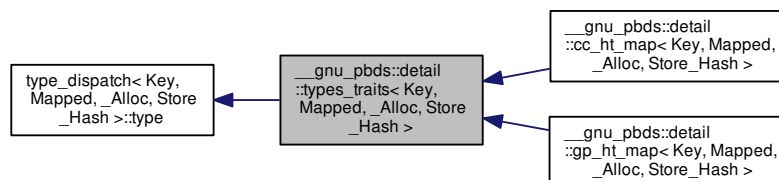
Definition at line 256 of file types_traits.hpp.

The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

5.351 __gnu_pbds::detail::types_traits< Key, Mapped, _Alloc, Store_Hash > Struct Template Reference

Inheritance diagram for __gnu_pbds::detail::types_traits< Key, Mapped, _Alloc, Store_Hash >:



Public Types

- typedef `std::pair< size_type, size_type >` **comp_hash**
- typedef `__rebind_a::const_pointer` **key_const_pointer**
- typedef `__rebind_a::const_reference` **key_const_reference**
- typedef `__rebind_a::pointer` **key_pointer**
- typedef `__rebind_a::reference` **key_reference**
- typedef `__rebind_a::value_type` **key_type**
- typedef `__nothrowcopy::indicator` **no_throw_indicator**
- typedef `_Alloc::size_type` **size_type**
- typedef `integral_constant< int, Store_Hash >` **store_extra**

Public Attributes

- no_throw_indicator **m_no_throw_copies_indicator**
- store_extra **m_store_extra_indicator**

5.351.1 Detailed Description

```
template<typename Key, typename Mapped, typename _Alloc, bool Store_Hash>
struct __gnu_pbds::detail::types_traits< Key, Mapped, _Alloc, Store_Hash >
```

Traits for abstract types.

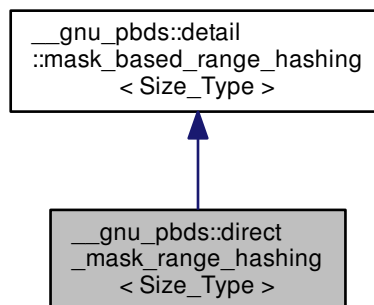
Definition at line 263 of file `types_traits.hpp`.

The documentation for this struct was generated from the following file:

- [types_traits.hpp](#)

5.352 `__gnu_pbds::direct_mask_range_hashing< Size_Type >` Class Template Reference

Inheritance diagram for `__gnu_pbds::direct_mask_range_hashing< Size_Type >`:



Public Types

- typedef Size_Type **size_type**

Public Member Functions

- void **swap** ([direct_mask_range_hashing](#)< Size_Type > &other)

Protected Member Functions

- void **notify_resized** (size_type size)
- size_type [operator\(\)](#) (size_type hash) const
- size_type **range_hash** (size_type hash) const
- void **swap** (mask_based_range_hashing &other)

5.352.1 Detailed Description

```
template<typename Size_Type = std::size_t>
class __gnu_pbds::direct_mask_range_hashing< Size_Type >
```

A mask range-hashing class (uses a bitmask).

Definition at line 109 of file hash_policy.hpp.

5.352.2 Member Function Documentation

5.352.2.1 `template<typename Size_Type > direct_mask_range_hashing< Size_Type >::size_type
__gnu_pbds::direct_mask_range_hashing< Size_Type >::operator() (size_type hash) const [inline,
[protected]`

Transforms the __hash value hash into a ranged-hash value (using a bit-mask).

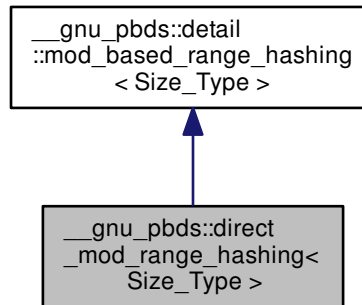
Definition at line 57 of file hash_policy.hpp.

The documentation for this class was generated from the following file:

- [hash_policy.hpp](#)

5.353 `__gnu_pbds::direct_mod_range_hashing< Size_Type >` Class Template Reference

Inheritance diagram for `__gnu_pbds::direct_mod_range_hashing< Size_Type >`:



Public Types

- typedef `Size_Type` **size_type**

Public Member Functions

- void **swap** (`direct_mod_range_hashing< Size_Type > &other`)

Protected Member Functions

- void **notify_resized** (`size_type size`)
- `size_type` **operator()** (`size_type hash`) const
- `size_type` **range_hash** (`size_type s`) const
- void **swap** (`mod_based_range_hashing &other`)

5.353.1 Detailed Description

```
template<typename Size_Type = std::size_t>
class __gnu_pbds::direct_mod_range_hashing< Size_Type >
```

A mod range-hashing class (uses the modulo function).

Definition at line 141 of file `hash_policy.hpp`.

5.353.2 Member Function Documentation

5.353.2.1 `template<typename Size_Type > direct_mod_range_hashing< Size_Type >::size_type
__gnu_pbds::direct_mod_range_hashing< Size_Type >::operator() (size_type hash) const` `[inline],`
`[protected]`

Transforms the `__hash` value `hash` into a ranged-hash value (using a modulo operation).

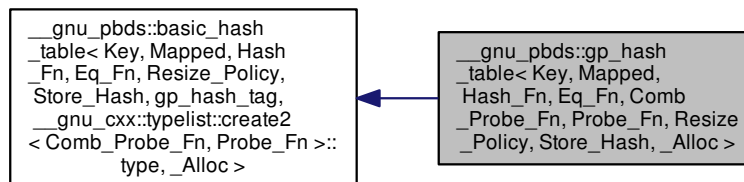
Definition at line 57 of file `hash_policy.hpp`.

The documentation for this class was generated from the following file:

- [hash_policy.hpp](#)

5.354 `__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >` Class Template Reference

Inheritance diagram for `__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >`:



Public Types

- typedef `Comb_Probe_Fn` **`comb_probe_fn`**
- typedef [gp_hash_tag](#) **`container_category`**
- typedef `Eq_Fn` **`eq_fn`**
- typedef `Hash_Fn` **`hash_fn`**
- typedef `Probe_Fn` **`probe_fn`**
- typedef `Resize_Policy` **`resize_policy`**

Public Member Functions

- [gp_hash_table](#) ()
- [gp_hash_table](#) (const hash_fn &h)
- [gp_hash_table](#) (const hash_fn &h, const eq_fn &e)
- [gp_hash_table](#) (const hash_fn &h, const eq_fn &e, const comb_probe_fn &cp)
- [gp_hash_table](#) (const hash_fn &h, const eq_fn &e, const comb_probe_fn &cp, const probe_fn &p)
- [gp_hash_table](#) (const hash_fn &h, const eq_fn &e, const comb_probe_fn &cp, const probe_fn &p, const resize_policy &rp)
- `template<typename It >`
[gp_hash_table](#) (It first, It last)
- `template<typename It >`
[gp_hash_table](#) (It first, It last, const hash_fn &h)
- `template<typename It >`
[gp_hash_table](#) (It first, It last, const hash_fn &h, const eq_fn &e)
- `template<typename It >`
[gp_hash_table](#) (It first, It last, const hash_fn &h, const eq_fn &e, const comb_probe_fn &cp)
- `template<typename It >`
[gp_hash_table](#) (It first, It last, const hash_fn &h, const eq_fn &e, const comb_probe_fn &cp, const probe_fn &p)
- `template<typename It >`
[gp_hash_table](#) (It first, It last, const hash_fn &h, const eq_fn &e, const comb_probe_fn &cp, const probe_fn &p, const resize_policy &rp)
- [gp_hash_table](#) (const [gp_hash_table](#) &other)
- [gp_hash_table](#) & **operator=** (const [gp_hash_table](#) &other)
- void **swap** ([gp_hash_table](#) &other)

5.354.1 Detailed Description

```
template<typename Key, typename Mapped, typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn
= typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn = detail::default_comb_hash_fn::type, typename Probe_Fn
= typename detail::default_probe_fn<Comb_Probe_Fn>::type, typename Resize_Policy = typename detail::default_resize_policy<
Comb_Probe_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>> >
class __gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >
```

A general-probing hash-based associative container.

Template Parameters

<i>Key</i>	Key type.
<i>Mapped</i>	Map type.
<i>Hash_Fn</i>	Hashing functor.
<i>Eq_Fn</i>	Equal functor.
<i>Comb_Probe_Fn</i>	Combining probe functor. If <i>Hash_Fn</i> is not null_type, then this is the ranged-probe functor; otherwise, this is the range-hashing functor. XXX See Design::Hash-Based Containers::Hash Policies.
<i>Probe_Fn</i>	Probe functor.
<i>Resize_Policy</i>	Resizes hash.
<i>Store_Hash</i>	Indicates whether the hash value will be stored along with each key. If <i>Hash_Fn</i> is null_type, then the container will not compile if this value is true
<i>_Alloc</i>	Allocator type.

Base tag choices are: `gp_hash_tag`.

Base is `basic_hash_table`.

Definition at line 368 of file `assoc_container.hpp`.

5.354.2 Constructor & Destructor Documentation

5.354.2.1 `template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn = detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_Probe_Fn>::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>> __gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >::gp_hash_table () [inline]`

Default constructor.

Definition at line 382 of file `assoc_container.hpp`.

5.354.2.2 `template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn = detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_Probe_Fn>::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>> __gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >::gp_hash_table (const hash_fn & h) [inline]`

Constructor taking some policy objects. `r_hash_fn` will be copied by the `hash_fn` object of the container object.

Definition at line 386 of file `assoc_container.hpp`.

5.354.2.3 `template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn = detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_Probe_Fn>::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>> __gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >::gp_hash_table (const hash_fn & h, const eq_fn & e) [inline]`

Constructor taking some policy objects. `r_hash_fn` will be copied by the `hash_fn` object of the container object, and `r_eq_fn` will be copied by the `eq_fn` object of the container object.

Definition at line 393 of file `assoc_container.hpp`.

```
5.354.2.4 template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type,
typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn =
detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_Probe_Fn>::type,
typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>::type, bool Store_Hash =
detail::default_store_hash, typename _Alloc = std::allocator<char>> __gnu_pbds::gp_hash_table< Key, Mapped,
Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >::gp_hash_table ( const hash_fn &
h, const eq_fn & e, const comb_probe_fn & cp ) [inline]
```

Constructor taking some policy objects. `r_hash_fn` will be copied by the `hash_fn` object of the container object, `r_eq_fn` will be copied by the `eq_fn` object of the container object, and `r_comb_probe_fn` will be copied by the `comb_probe_fn` object of the container object.

Definition at line 401 of file `assoc_container.hpp`.

```
5.354.2.5 template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type,
typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn =
detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_Probe_Fn>::type,
typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>::type, bool Store_Hash =
detail::default_store_hash, typename _Alloc = std::allocator<char>> __gnu_pbds::gp_hash_table< Key, Mapped,
Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >::gp_hash_table ( const hash_fn &
h, const eq_fn & e, const comb_probe_fn & cp, const probe_fn & p ) [inline]
```

Constructor taking some policy objects. `r_hash_fn` will be copied by the `hash_fn` object of the container object, `r_eq_fn` will be copied by the `eq_fn` object of the container object, `r_comb_probe_fn` will be copied by the `comb_probe_fn` object of the container object, and `r_probe_fn` will be copied by the `probe_fn` object of the container object.

Definition at line 410 of file `assoc_container.hpp`.

```
5.354.2.6 template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type,
typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn =
detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_Probe_Fn>::type,
typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>::type, bool Store_Hash =
detail::default_store_hash, typename _Alloc = std::allocator<char>> __gnu_pbds::gp_hash_table< Key, Mapped,
Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >::gp_hash_table ( const hash_fn &
h, const eq_fn & e, const comb_probe_fn & cp, const probe_fn & p, const resize_policy & rp ) [inline]
```

Constructor taking some policy objects. `r_hash_fn` will be copied by the `hash_fn` object of the container object, `r_eq_fn` will be copied by the `eq_fn` object of the container object, `r_comb_probe_fn` will be copied by the `comb_probe_fn` object of the container object, `r_probe_fn` will be copied by the `probe_fn` object of the container object, and `r_resize_policy` will be copied by the `Resize_Policy` object of the container object.

Definition at line 422 of file `assoc_container.hpp`.

```
5.354.2.7 template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type,
typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn =
detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_Probe_Fn>::type,
typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>::type, bool Store_Hash
= detail::default_store_hash, typename _Alloc = std::allocator<char>> template<typename It >
__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy,
Store_Hash, _Alloc >::gp_hash_table ( It first, It last ) [inline]
```

Constructor taking `__iterators` to a range of `value_types`. The `value_types` between `first_it` and `last_it` will be inserted into the container object.

Definition at line 430 of file `assoc_container.hpp`.

5.354.2.8 `template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn = detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_Probe_Fn>::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>> template<typename It > __gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >::gp_hash_table (It first, It last, const hash_fn & h) [inline]`

Constructor taking __iterators to a range of value_types and some policy objects. The value_types between first_it and last_it will be inserted into the container object. r_hash_fn will be copied by the hash_fn object of the container object.

Definition at line 438 of file assoc_container.hpp.

5.354.2.9 `template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn = detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_Probe_Fn>::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>> template<typename It > __gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >::gp_hash_table (It first, It last, const hash_fn & h, const eq_fn & e) [inline]`

Constructor taking __iterators to a range of value_types and some policy objects. The value_types between first_it and last_it will be inserted into the container object. r_hash_fn will be copied by the hash_fn object of the container object, and r_eq_fn will be copied by the eq_fn object of the container object.

Definition at line 449 of file assoc_container.hpp.

5.354.2.10 `template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn = detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_Probe_Fn>::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>> template<typename It > __gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >::gp_hash_table (It first, It last, const hash_fn & h, const eq_fn & e, const comb_probe_fn & cp) [inline]`

Constructor taking __iterators to a range of value_types and some policy objects. The value_types between first_it and last_it will be inserted into the container object. r_hash_fn will be copied by the hash_fn object of the container object, r_eq_fn will be copied by the eq_fn object of the container object, and r_comb_probe_fn will be copied by the comb_probe_fn object of the container object.

Definition at line 461 of file assoc_container.hpp.

5.354.2.11 `template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type, typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn = detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_Probe_Fn>::type, typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>::type, bool Store_Hash = detail::default_store_hash, typename _Alloc = std::allocator<char>> template<typename It > __gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy, Store_Hash, _Alloc >::gp_hash_table (It first, It last, const hash_fn & h, const eq_fn & e, const comb_probe_fn & cp, const probe_fn & p) [inline]`

Constructor taking __iterators to a range of value_types and some policy objects. The value_types between first_it and last_it will be inserted into the container object. r_hash_fn will be copied by the hash_fn object of the container object, r_eq_fn will be copied by the eq_fn object of the container object, r_comb_probe_fn will be copied by the comb_probe_fn object of the container object, and r_probe_fn will be copied by the probe_fn object of the container object.

Definition at line 475 of file assoc_container.hpp.

```

5.354.2.12 template<typename Key , typename Mapped , typename Hash_Fn = typename detail::default_hash_fn<Key>::type,
typename Eq_Fn = typename detail::default_eq_fn<Key>::type, typename Comb_Probe_Fn =
detail::default_comb_hash_fn::type, typename Probe_Fn = typename detail::default_probe_fn<Comb_Probe_Fn>::type,
typename Resize_Policy = typename detail::default_resize_policy<Comb_Probe_Fn>::type, bool Store_Hash
= detail::default_store_hash, typename _Alloc = std::allocator<char>>> template<typename It >
__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_Policy,
Store_Hash, _Alloc >::gp_hash_table ( It first, It last, const hash_fn & h, const eq_fn & e, const comb_probe_fn &
cp, const probe_fn & p, const resize_policy & rp ) [inline]

```

Constructor taking __iterators to a range of value_types and some policy objects. The value_types between first_it and last_it will be inserted into the container object. r_hash_fn will be copied by the hash_fn object of the container object, r_eq_fn will be copied by the eq_fn object of the container object, r_comb_probe_fn will be copied by the comb_←probe_fn object of the container object, r_probe_fn will be copied by the probe_fn object of the container object, and r_resize_policy will be copied by the resize_policy object of the container object.

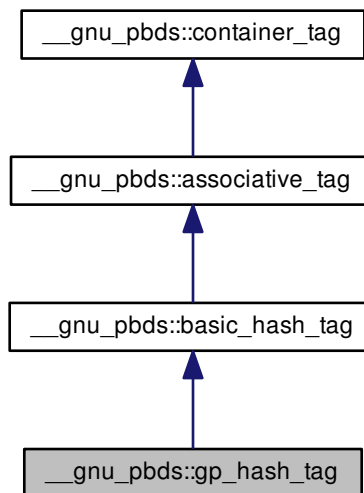
Definition at line 491 of file assoc_container.hpp.

The documentation for this class was generated from the following file:

- [assoc_container.hpp](#)

5.355 __gnu_pbds::gp_hash_tag Struct Reference

Inheritance diagram for __gnu_pbds::gp_hash_tag:



5.355.1 Detailed Description

General-probing hash.

Definition at line 144 of file tag_and_trait.hpp.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.356 __gnu_pbds::hash_exponential_size_policy< Size_Type > Class Template Reference

Public Types

- typedef Size_Type **size_type**

Public Member Functions

- [hash_exponential_size_policy](#) (size_type start_size=8, size_type grow_factor=2)
- void **swap** ([hash_exponential_size_policy](#)< Size_Type > &other)

Protected Member Functions

- size_type **get_nearest_larger_size** (size_type size) const
- size_type **get_nearest_smaller_size** (size_type size) const

5.356.1 Detailed Description

```
template<typename Size_Type = std::size_t>
class __gnu_pbds::hash_exponential_size_policy< Size_Type >
```

A size policy whose sequence of sizes form an exponential sequence (typically powers of 2).

Definition at line 413 of file hash_policy.hpp.

5.356.2 Constructor & Destructor Documentation

5.356.2.1 `template<typename Size_Type > __gnu_pbds::hash_exponential_size_policy< Size_Type >::hash_exponential_size_policy (size_type start_size = 8, size_type grow_factor = 2)`

Default constructor, or onstructor taking a start_size, or constructor taking a start size and grow_factor. The policy will use the sequence of sizes start_size, start_size* grow_factor, start_size* grow_factor^2, ...

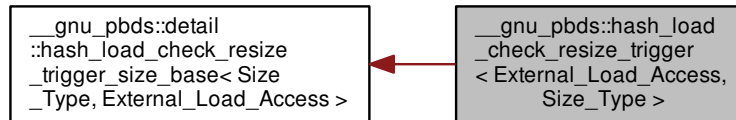
Definition at line 44 of file hash_policy.hpp.

The documentation for this class was generated from the following file:

- [hash_policy.hpp](#)

5.357 `__gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >` Class Template Reference

Inheritance diagram for `__gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >`:



Public Types

- enum { [external_load_access](#) }
- typedef `Size_Type` **size_type**

Public Member Functions

- [hash_load_check_resize_trigger](#) (float load_min=0.125, float load_max=0.5)
- `std::pair< float, float >` [get_loads](#) () const
- void [set_loads](#) (`std::pair< float, float >` load_pair)
- void **swap** ([hash_load_check_resize_trigger](#) &other)

Protected Member Functions

- bool **is_grow_needed** (size_type size, size_type num_entries) const
- bool **is_resize_needed** () const
- void [notify_cleared](#) ()
- void **notify_erase_search_collision** ()
- void **notify_erase_search_end** ()
- void **notify_erase_search_start** ()
- void **notify_erased** (size_type num_entries)
- void **notify_externally_resized** (size_type new_size)
- void **notify_find_search_collision** ()
- void **notify_find_search_end** ()
- void **notify_find_search_start** ()
- void **notify_insert_search_collision** ()
- void **notify_insert_search_end** ()
- void **notify_insert_search_start** ()
- void [notify_inserted](#) (size_type num_entries)
- void [notify_resized](#) (size_type new_size)

5.357.1 Detailed Description

```
template<bool External_Load_Access = false, typename Size_Type = std::size_t>
class __gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >
```

A resize trigger policy based on a load check. It keeps the load factor between some load factors `load_min` and `load_max`.

Definition at line 175 of file `hash_policy.hpp`.

5.357.2 Member Enumeration Documentation

5.357.2.1 `template<bool External_Load_Access = false, typename Size_Type = std::size_t> anonymous enum`

Enumerator

external_load_access Specifies whether the load factor can be accessed externally. The two options have different trade-offs in terms of flexibility, genericity, and encapsulation.

Definition at line 180 of file `hash_policy.hpp`.

5.357.3 Constructor & Destructor Documentation

5.357.3.1 `template<bool External_Load_Access, typename Size_Type > __gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >::hash_load_check_resize_trigger (float load_min = 0.125, float load_max = 0.5)`

Default constructor, or constructor taking `load_min` and `load_max` load factors between which this policy will keep the actual load.

Definition at line 47 of file `hash_policy.hpp`.

5.357.4 Member Function Documentation

5.357.4.1 `template<bool External_Load_Access, typename Size_Type > std::pair< float, float > __gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >::get_loads () const [inline]`

Returns a pair of the minimal and maximal loads, respectively.

Definition at line 236 of file `hash_policy.hpp`.

5.357.4.2 `template<bool External_Load_Access, typename Size_Type > void __gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >::notify_cleared () [protected]`

Notifies the table was cleared.

Definition at line 206 of file `hash_policy.hpp`.

5.357.4.3 `template<bool External_Load_Access, typename Size_Type > void __gnu_pbds::hash_load_check_↔
resize_trigger< External_Load_Access, Size_Type >::notify_inserted (size_type num_entries) [inline],
[protected]`

Notifies an element was inserted. The total number of entries in the table is `num_entries`.

Definition at line 109 of file `hash_policy.hpp`.

5.357.4.4 `template<bool External_Load_Access, typename Size_Type > void __gnu_pbds::hash_load_↔
check_resize_trigger< External_Load_Access, Size_Type >::notify_resized (size_type new_size)
[protected]`

Notifies the table was resized as a result of this object's signifying that a resize is needed.

Definition at line 151 of file `hash_policy.hpp`.

5.357.4.5 `template<bool External_Load_Access, typename Size_Type > void __gnu_pbds::hash_load_check_↔
_resize_trigger< External_Load_Access, Size_Type >::set_loads (std::pair< float, float > load_pair
)`

Sets the loads through a pair of the minimal and maximal loads, respectively.

Definition at line 245 of file `hash_policy.hpp`.

The documentation for this class was generated from the following file:

- [hash_policy.hpp](#)

5.358 `__gnu_pbds::hash_prime_size_policy` Class Reference

Public Types

- typedef std::size_t [size_type](#)

Public Member Functions

- [hash_prime_size_policy](#) ([size_type](#) start_size=8)
- void **swap** ([hash_prime_size_policy](#) &other)

Protected Member Functions

- [size_type](#) **get_nearest_larger_size** ([size_type](#) size) const
- [size_type](#) **get_nearest_smaller_size** ([size_type](#) size) const

5.358.1 Detailed Description

A size policy whose sequence of sizes form a nearly-exponential sequence of primes.

Definition at line 450 of file hash_policy.hpp.

5.358.2 Member Typedef Documentation

5.358.2.1 typedef std::size_t __gnu_pbds::hash_prime_size_policy::size_type

Size type.

Definition at line 454 of file hash_policy.hpp.

5.358.3 Constructor & Destructor Documentation

5.358.3.1 __gnu_pbds::hash_prime_size_policy::hash_prime_size_policy (size_type start_size = 8) [inline]

Default constructor, or onstructor taking a start_size The policy will use the sequence of sizes approximately start_size, start_size* 2, start_size* 2^2, ...

Definition at line 127 of file hash_policy.hpp.

The documentation for this class was generated from the following file:

- [hash_policy.hpp](#)

5.359 __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type > Class Template Reference

Inherits Size_Policy, and Trigger_Policy.

Public Types

- enum { **external_size_access** }
- typedef Size_Policy **size_policy**
- typedef Size_Type **size_type**
- typedef Trigger_Policy **trigger_policy**

Public Member Functions

- `hash_standard_resize_policy` ()
- `hash_standard_resize_policy` (const `Size_Policy` &`r_size_policy`)
- `hash_standard_resize_policy` (const `Size_Policy` &`r_size_policy`, const `Trigger_Policy` &`r_trigger_policy`)
- `size_type` `get_actual_size` () const
- `Size_Policy` & `get_size_policy` ()
- const `Size_Policy` & `get_size_policy` () const
- `Trigger_Policy` & `get_trigger_policy` ()
- const `Trigger_Policy` & `get_trigger_policy` () const
- void `resize` (`size_type` `suggested_new_size`)
- void `swap` (`hash_standard_resize_policy`< `Size_Policy`, `Trigger_Policy`, `External_Size_Access`, `Size_Type` > &`other`)

Protected Member Functions

- `size_type` `get_new_size` (`size_type` `size`, `size_type` `num_used_e`) const
- bool `is_resize_needed` () const
- void `notify_cleared` ()
- void `notify_erase_search_collision` ()
- void `notify_erase_search_end` ()
- void `notify_erase_search_start` ()
- void `notify_erased` (`size_type` `num_e`)
- void `notify_find_search_collision` ()
- void `notify_find_search_end` ()
- void `notify_find_search_start` ()
- void `notify_insert_search_collision` ()
- void `notify_insert_search_end` ()
- void `notify_insert_search_start` ()
- void `notify_inserted` (`size_type` `num_e`)
- void `notify_resized` (`size_type` `new_size`)

5.359.1 Detailed Description

```
template<typename Size_Policy = hash_exponential_size_policy<>, typename Trigger_Policy = hash_load_check_resize_↵
trigger<>, bool External_Size_Access = false, typename Size_Type = std::size_t>
class __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type >
```

A resize policy which delegates operations to size and trigger policies.

Definition at line 489 of file `hash_policy.hpp`.

5.359.2 Constructor & Destructor Documentation

5.359.2.1 `template<typename Size_Policy , typename Trigger_Policy , bool External_Size_Access, typename Size_Type > __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type >::hash_standard_resize_policy ()`

Default constructor.

Definition at line 44 of file `hash_policy.hpp`.

```
5.359.2.2  template<typename Size_Policy, typename Trigger_Policy , bool External_Size_Access, typename Size_Type >
            __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type
            >::hash_standard_resize_policy ( const Size_Policy & r_size_policy )
```

constructor taking some policies `r_size_policy` will be copied by the `Size_Policy` object of this object.

Definition at line 50 of file `hash_policy.hpp`.

```
5.359.2.3  template<typename Size_Policy, typename Trigger_Policy, bool External_Size_Access, typename Size_Type >
            __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type
            >::hash_standard_resize_policy ( const Size_Policy & r_size_policy, const Trigger_Policy & r_trigger_policy )
```

constructor taking some policies. `r_size_policy` will be copied by the `Size_Policy` object of this object. `r_trigger_policy` will be copied by the `Trigger_Policy` object of this object.

Definition at line 56 of file `hash_policy.hpp`.

5.359.3 Member Function Documentation

```
5.359.3.1  template<typename Size_Policy , typename Trigger_Policy , bool External_Size_Access, typename Size_Type >
            hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type >::size_type
            __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type
            >::get_actual_size ( ) const    [inline]
```

Returns the actual size of the container.

Definition at line 177 of file `hash_policy.hpp`.

```
5.359.3.2  template<typename Size_Policy , typename Trigger_Policy , bool External_Size_Access, typename Size_Type >
            hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type >::size_type
            __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type
            >::get_new_size ( size_type size, size_type num_used_e ) const    [protected]
```

Queries what the new size should be, when the container is resized naturally. The current `__size` of the container is `size`, and the number of used entries within the container is `num_used_e`.

Definition at line 158 of file `hash_policy.hpp`.

```
5.359.3.3  template<typename Size_Policy , typename Trigger_Policy , bool External_Size_Access, typename Size_Type >
            Size_Policy & __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access,
            Size_Type >::get_size_policy ( )
```

Access to the `Size_Policy` object used.

Definition at line 242 of file `hash_policy.hpp`.

5.359.3.4 `template<typename Size_Policy , typename Trigger_Policy , bool External_Size_Access, typename Size_Type > const Size_Policy & __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type >::get_size_policy () const`

Const access to the `Size_Policy` object used.

Definition at line 248 of file `hash_policy.hpp`.

5.359.3.5 `template<typename Size_Policy , typename Trigger_Policy , bool External_Size_Access, typename Size_Type > Trigger_Policy & __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type >::get_trigger_policy ()`

Access to the `Trigger_Policy` object used.

Definition at line 230 of file `hash_policy.hpp`.

5.359.3.6 `template<typename Size_Policy , typename Trigger_Policy , bool External_Size_Access, typename Size_Type > const Trigger_Policy & __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type >::get_trigger_policy () const`

Access to the `Trigger_Policy` object used.

Definition at line 236 of file `hash_policy.hpp`.

5.359.3.7 `template<typename Size_Policy , typename Trigger_Policy , bool External_Size_Access, typename Size_Type > void __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type >::resize (size_type suggested_new_size)`

Resizes the container to `suggested_new_size`, a suggested size (the actual size will be determined by the `Size_Policy` object).

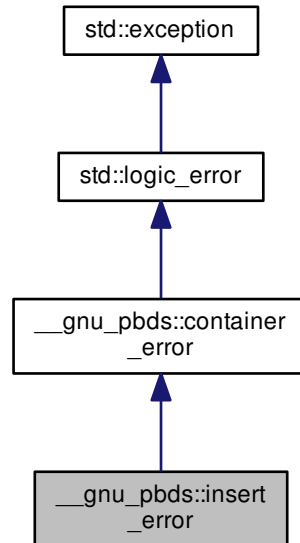
Definition at line 186 of file `hash_policy.hpp`.

The documentation for this class was generated from the following file:

- [hash_policy.hpp](#)

5.360 `__gnu_pbds::insert_error` Struct Reference

Inheritance diagram for `__gnu_pbds::insert_error`:



Public Member Functions

- virtual const char * [what](#) () const `_GLIBCXX_TXN_SAFE_DYN` noexcept

5.360.1 Detailed Description

An entry cannot be inserted into a container object for logical reasons (not, e.g., if memory is unavailable, in which case the `allocator_type`'s exception will be thrown).

Definition at line 66 of file `exception.hpp`.

5.360.2 Member Function Documentation

5.360.2.1 virtual const char* `std::logic_error::what` () const `[virtual]`, `[noexcept]`, `[inherited]`

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

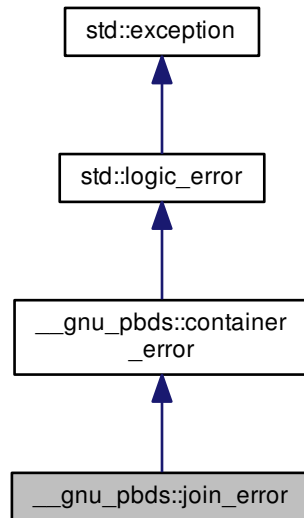
Reimplemented in [std::future_error](#).

The documentation for this struct was generated from the following file:

- [exception.hpp](#)

5.361 `__gnu_pbds::join_error` Struct Reference

Inheritance diagram for `__gnu_pbds::join_error`:

**Public Member Functions**

- virtual const char * [what](#) () const `_GLIBCXX_TXN_SAFE_DYN` noexcept

5.361.1 Detailed Description

A join cannot be performed logical reasons (i.e., the ranges of the two container objects being joined overlaps.

Definition at line 70 of file `exception.hpp`.

5.361.2 Member Function Documentation

5.361.2.1 virtual const char* `std::logic_error::what` () const `[virtual]`, `[noexcept]`, `[inherited]`

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

Reimplemented in [std::future_error](#).

The documentation for this struct was generated from the following file:

- [exception.hpp](#)

5.362 `__gnu_pbds::linear_probe_fn< Size_Type >` Class Template Reference

Public Types

- typedef `Size_Type` **size_type**

Public Member Functions

- void **swap** ([linear_probe_fn< Size_Type >](#) &other)

Protected Member Functions

- `size_type` [operator\(\)](#) (`size_type` i) const

5.362.1 Detailed Description

```
template<typename Size_Type = std::size_t>
class __gnu_pbds::linear_probe_fn< Size_Type >
```

A probe sequence policy using fixed increments.

Definition at line 61 of file `hash_policy.hpp`.

5.362.2 Member Function Documentation

5.362.2.1 `template<typename Size_Type > linear_probe_fn< Size_Type >::size_type __gnu_pbds::linear_probe_fn< Size_Type >::operator() (size_type i) const` `[inline]`, `[protected]`

Returns the i-th offset from the hash value.

Definition at line 51 of file `hash_policy.hpp`.

The documentation for this class was generated from the following file:

- [hash_policy.hpp](#)

5.363 `__gnu_pbds::list_update< Key, Mapped, Eq_Fn, Update_Policy, _Alloc >` Class Template Reference

Inherits type< `Key`, `Mapped`, `_Alloc`, `list_update_tag`, `__gnu_cxx::typelist::create2< Eq_Fn, Update_Policy >::type` >.

Public Types

- typedef `list_update_tag` `container_category`
- typedef `Eq_Fn` `eq_fn`
- typedef `Update_Policy` `update_policy`

Public Member Functions

- template<typename `It` >
`list_update` (`It` `first`, `It` `last`)
- `list_update` (const `list_update` &`other`)
- `list_update` & `operator=` (const `list_update` &`other`)
- void `swap` (`list_update` &`other`)

5.363.1 Detailed Description

```
template<typename Key, typename Mapped, class Eq_Fn = typename detail::default_eq_fn<Key>::type, class Update_Policy =
detail::default_update_policy::type, class _Alloc = std::allocator<char>>>
class __gnu_pbds::list_update< Key, Mapped, Eq_Fn, Update_Policy, _Alloc >
```

A list-update based associative container.

Template Parameters

<i>Key</i>	Key type.
<i>Mapped</i>	Map type.
<i>Eq_Fn</i>	Equal functor.
<i>Update_Policy</i>	Update policy, determines when an element will be moved to the front of the list.
<i>_Alloc</i>	Allocator type.

Base is `detail::lu_map`.

Definition at line 815 of file `assoc_container.hpp`.

5.363.2 Constructor & Destructor Documentation

5.363.2.1 `template<typename Key , typename Mapped , class Eq_Fn = typename detail::default_eq_fn<Key>::type, class Update_Policy = detail::default_update_policy::type, class _Alloc = std::allocator<char>>> template<typename It > __gnu_pbds::list_update< Key, Mapped, Eq_Fn, Update_Policy, _Alloc >::list_update (It first, It last)`
[`inline`]

Constructor taking `__iterators` to a range of `value_types`. The `value_types` between `first_it` and `last_it` will be inserted into the container object.

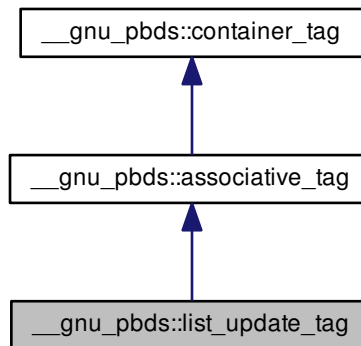
Definition at line 831 of file `assoc_container.hpp`.

The documentation for this class was generated from the following file:

- [assoc_container.hpp](#)

5.364 __gnu_pbds::list_update_tag Struct Reference

Inheritance diagram for __gnu_pbds::list_update_tag:



5.364.1 Detailed Description

List-update.

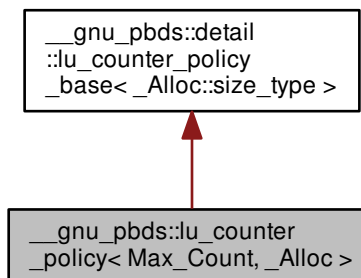
Definition at line 168 of file tag_and_trait.hpp.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.365 __gnu_pbds::lu_counter_policy< Max_Count, _Alloc > Class Template Reference

Inheritance diagram for __gnu_pbds::lu_counter_policy< Max_Count, _Alloc >:



Public Types

- enum { `max_count` }
- typedef `_Alloc allocator_type`
- typedef `__rebind_m::other::reference metadata_reference`
- typedef `detail::lu_counter_metadata< size_type > metadata_type`
- typedef `allocator_type::size_type size_type`

Public Member Functions

- `metadata_type operator() ()` const
- `bool operator() (metadata_reference r_data)` const

Private Member Functions

- `lu_counter_metadata< size_type > operator() (size_type max_size)` const
- `bool operator() (Metadata_Reference r_data, size_type m_max_count)` const

5.365.1 Detailed Description

```
template<std::size_t Max_Count = 5, typename _Alloc = std::allocator<char>>
class __gnu_pbds::lu_counter_policy< Max_Count, _Alloc >
```

A list-update policy that moves elements to the front of the list based on the counter algorithm.

Definition at line 92 of file `list_update_policy.hpp`.

5.365.2 Member Typedef Documentation

5.365.2.1 `template<std::size_t Max_Count = 5, typename _Alloc = std::allocator<char>> typedef __rebind_m::other::reference __gnu_pbds::lu_counter_policy< Max_Count, _Alloc >::metadata_reference`

Reference to metadata on which this functor operates.

Definition at line 115 of file `list_update_policy.hpp`.

5.365.2.2 `template<std::size_t Max_Count = 5, typename _Alloc = std::allocator<char>> typedef detail::lu_counter_metadata<size_type> __gnu_pbds::lu_counter_policy< Max_Count, _Alloc >::metadata_type`

Metadata on which this functor operates.

Definition at line 107 of file `list_update_policy.hpp`.

5.365.3 Member Enumeration Documentation

5.365.3.1 `template<std::size_t Max_Count = 5, typename _Alloc = std::allocator<char>> anonymous enum`

Enumerator

`max_count` When some element is accessed this number of times, it will be moved to the front of the list.

Definition at line 99 of file `list_update_policy.hpp`.

5.365.4 Member Function Documentation

5.365.4.1 `template<std::size_t Max_Count = 5, typename _Alloc = std::allocator<char>> metadata_type __gnu_pbds::lu_counter_policy< Max_Count, _Alloc >::operator() () const [inline]`

Creates a metadata object.

Definition at line 119 of file `list_update_policy.hpp`.

5.365.4.2 `template<std::size_t Max_Count = 5, typename _Alloc = std::allocator<char>> bool __gnu_pbds::lu_counter_policy< Max_Count, _Alloc >::operator() (metadata_reference r_data) const [inline]`

Decides whether a metadata object should be moved to the front of the list.

Definition at line 125 of file `list_update_policy.hpp`.

The documentation for this class was generated from the following file:

- [list_update_policy.hpp](#)

5.366 `__gnu_pbds::lu_move_to_front_policy< _Alloc >` Class Template Reference

Public Types

- typedef `_Alloc` **`allocator_type`**
- typedef `__rebind_m::other::reference` [metadata_reference](#)
- typedef [null_type](#) [metadata_type](#)

Public Member Functions

- [metadata_type operator\(\) \(\) const](#)
- [bool operator\(\) \(metadata_reference r_metadata\) const](#)

5.366.1 Detailed Description

```
template<typename _Alloc = std::allocator<char>>
class __gnu_pbds::lu_move_to_front_policy<_Alloc>
```

A list-update policy that unconditionally moves elements to the front of the list. A null type means that each link in a list-based container does not actually need metadata.

Definition at line 57 of file `list_update_policy.hpp`.

5.366.2 Member Typedef Documentation

5.366.2.1 `template<typename _Alloc = std::allocator<char>> typedef __rebind_m::other::reference
__gnu_pbds::lu_move_to_front_policy<_Alloc>::metadata_reference`

Reference to metadata on which this functor operates.

Definition at line 70 of file `list_update_policy.hpp`.

5.366.2.2 `template<typename _Alloc = std::allocator<char>> typedef null_type __gnu_pbds::lu_move_to_front_
policy<_Alloc>::metadata_type`

Metadata on which this functor operates.

Definition at line 63 of file `list_update_policy.hpp`.

5.366.3 Member Function Documentation

5.366.3.1 `template<typename _Alloc = std::allocator<char>> metadata_type __gnu_pbds::lu_move_to_front_policy<
_Alloc>::operator()() const [inline]`

Creates a metadata object.

Definition at line 74 of file `list_update_policy.hpp`.

5.366.3.2 `template<typename _Alloc = std::allocator<char>> bool __gnu_pbds::lu_move_to_front_policy<_Alloc
>::operator()(metadata_reference r_metadata) const [inline]`

Decides whether a metadata object should be moved to the front of the list.

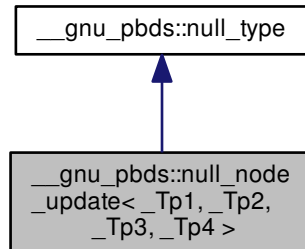
Definition at line 80 of file `list_update_policy.hpp`.

The documentation for this class was generated from the following file:

- [list_update_policy.hpp](#)

5.367 `__gnu_pbds::null_node_update<_Tp1, _Tp2, _Tp3, _Tp4>` Struct Template Reference

Inheritance diagram for `__gnu_pbds::null_node_update<_Tp1, _Tp2, _Tp3, _Tp4>`:



5.367.1 Detailed Description

```
template<typename _Tp1, typename _Tp2, typename _Tp3, typename _Tp4>
struct __gnu_pbds::null_node_update<_Tp1, _Tp2, _Tp3, _Tp4>
```

A null node updator, indicating that no node updates are required.

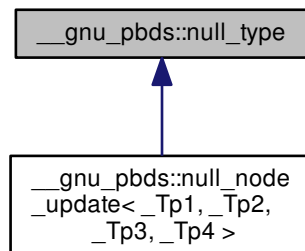
Definition at line 214 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.368 `__gnu_pbds::null_type` Struct Reference

Inheritance diagram for `__gnu_pbds::null_type`:



5.368.1 Detailed Description

Represents no type, or absence of type, for template tricks.

In a mapped-policy, indicates that an associative container is a set.

In a list-update policy, indicates that each link does not need metadata.

In a hash policy, indicates that the combining hash function is actually a ranged hash function.

In a probe policy, indicates that the combining probe function is actually a ranged probe function.

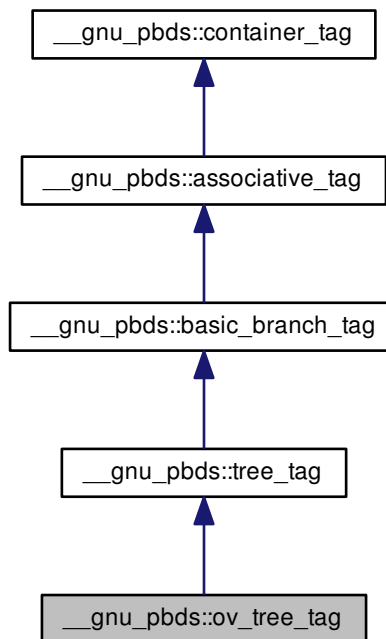
Definition at line 210 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.369 `__gnu_pbds::ov_tree_tag` Struct Reference

Inheritance diagram for `__gnu_pbds::ov_tree_tag`:



5.369.1 Detailed Description

Ordered-vector tree.

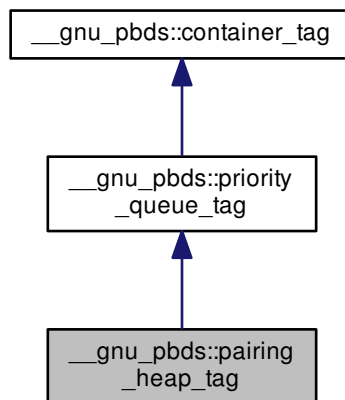
Definition at line 159 of file tag_and_trait.hpp.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.370 __gnu_pbds::pairing_heap_tag Struct Reference

Inheritance diagram for __gnu_pbds::pairing_heap_tag:



5.370.1 Detailed Description

Pairing-heap.

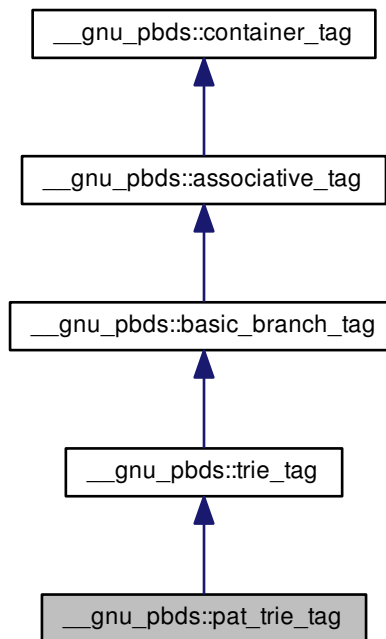
Definition at line 174 of file tag_and_trait.hpp.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.371 __gnu_pbds::pat_trie_tag Struct Reference

Inheritance diagram for __gnu_pbds::pat_trie_tag:



5.371.1 Detailed Description

PATRICIA trie.

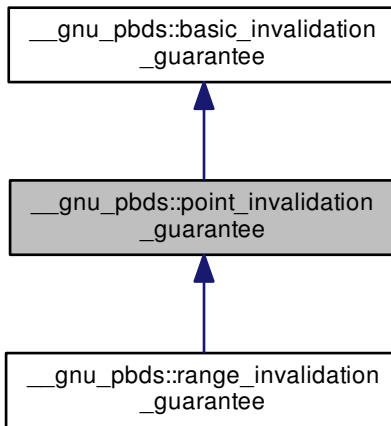
Definition at line 165 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.372 `__gnu_pbds::point_invalidation_guarantee` Struct Reference

Inheritance diagram for `__gnu_pbds::point_invalidation_guarantee`:



5.372.1 Detailed Description

Signifies an invalidation guarantee that includes all those of its base, and additionally, that any point-type iterator, pointer, or reference to a container object's mapped value type is valid as long as its corresponding entry has not be erased, regardless of modifications to the container object.

Definition at line 103 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.373 `__gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc>` Class Template Reference

Inherits `type<_Tv, Cmp_Fn, _Alloc, Tag>`.

Public Types

- typedef `_Alloc` **allocator_type**
- typedef `Cmp_Fn` **cmp_fn**
- typedef `base_type::const_iterator` **const_iterator**
- typedef `__rebind_va::const_pointer` **const_pointer**
- typedef `__rebind_va::const_reference` **const_reference**
- typedef `Tag` **container_category**
- typedef `allocator_type::difference_type` **difference_type**
- typedef `base_type::iterator` **iterator**
- typedef `base_type::point_const_iterator` **point_const_iterator**
- typedef `base_type::point_iterator` **point_iterator**
- typedef `__rebind_va::pointer` **pointer**
- typedef `__rebind_va::reference` **reference**
- typedef `allocator_type::size_type` **size_type**
- typedef `_Tv` **value_type**

Public Member Functions

- [`priority_queue`](#) (const `cmp_fn` &`r_cmp_fn`)
- `template<typename It >`
[`priority_queue`](#) (It `first_it`, It `last_it`)
- `template<typename It >`
[`priority_queue`](#) (It `first_it`, It `last_it`, const `cmp_fn` &`r_cmp_fn`)
- **`priority_queue`** (const [`priority_queue`](#) &`other`)
- [`priority_queue`](#) & **`operator=`** (const [`priority_queue`](#) &`other`)
- void **`swap`** ([`priority_queue`](#) &`other`)

5.373.1 Detailed Description

```
template<typename _Tv, typename Cmp_Fn = std::less<_Tv>, typename Tag = pairing_heap_tag, typename _Alloc = std::allocator<char>>
class __gnu_pbds::priority_queue<_Tv, Cmp_Fn, Tag, _Alloc>
```

A priority queue composed of one specific heap policy.

Template Parameters

<code>_Tv</code>	Value type.
<code>Cmp_Fn</code>	Comparison functor.
<code>Tag</code>	Instantiating data structure type, see <code>container_tag</code> .
<code>_Alloc</code>	Allocator type.

Base is dispatched at compile time via `Tag`, from the following choices: `binary_heap_tag`, `binomial_heap_tag`, `pairing_heap_tag`, `rc_binomial_heap_tag`, `thin_heap_tag`

Base choices are: `detail::binary_heap`, `detail::binomial_heap`, `detail::pairing_heap`, `detail::rc_binomial_heap`, `detail::thin_heap`.

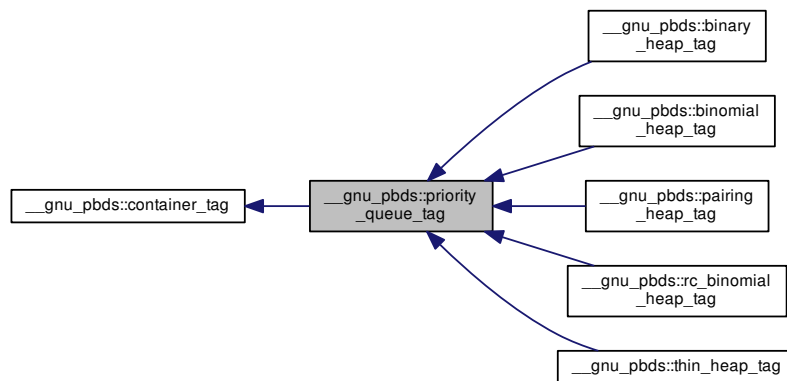
Definition at line 83 of file priority_queue.hpp.

The documentation for this class was generated from the following file:

- [priority_queue.hpp](#)

5.374 __gnu_pbds::priority_queue_tag Struct Reference

Inheritance diagram for __gnu_pbds::priority_queue_tag:



5.374.1 Detailed Description

Basic priority-queue.

Definition at line 171 of file tag_and_trait.hpp.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.375 __gnu_pbds::quadratic_probe_fn< Size_Type > Class Template Reference

Public Types

- typedef Size_Type **size_type**

Public Member Functions

- void **swap** ([quadratic_probe_fn](#)< Size_Type > &other)

Protected Member Functions

- `size_type operator() (size_type i) const`

5.375.1 Detailed Description

```
template<typename Size_Type = std::size_t>
class __gnu_pbds::quadratic_probe_fn< Size_Type >
```

A probe sequence policy using square increments.

Definition at line 85 of file `hash_policy.hpp`.

5.375.2 Member Function Documentation

5.375.2.1 `template<typename Size_Type > quadratic_probe_fn< Size_Type >::size_type __gnu_pbds::quadratic_probe_fn< Size_Type >::operator() (size_type i) const` `[inline]`, `[protected]`

Returns the i-th offset from the hash value.

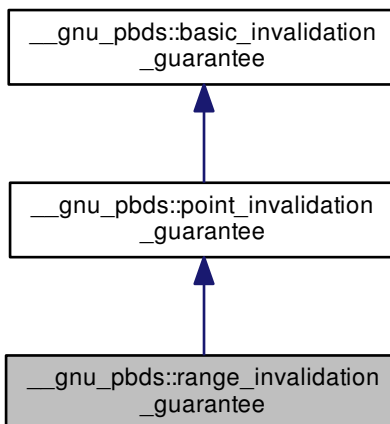
Definition at line 51 of file `hash_policy.hpp`.

The documentation for this class was generated from the following file:

- [hash_policy.hpp](#)

5.376 `__gnu_pbds::range_invalidation_guarantee` Struct Reference

Inheritance diagram for `__gnu_pbds::range_invalidation_guarantee`:



5.376.1 Detailed Description

Signifies an invalidation guarantee that includes all those of its base, and additionally, that any range-type iterator (including the returns of `begin()` and `end()`) is in the correct relative positions to other range-type iterators as long as its corresponding entry has not be erased, regardless of modifications to the container object.

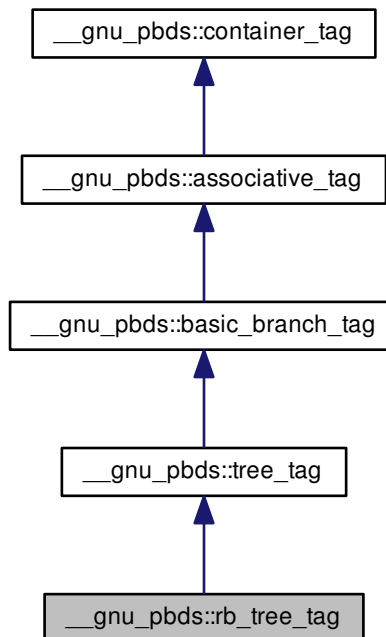
Definition at line 114 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.377 __gnu_pbds::rb_tree_tag Struct Reference

Inheritance diagram for `__gnu_pbds::rb_tree_tag`:



5.377.1 Detailed Description

Red-black tree.

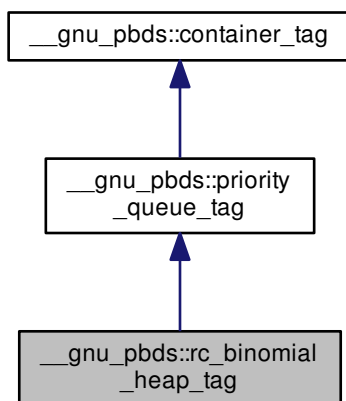
Definition at line 153 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.378 `__gnu_pbds::rc_binomial_heap_tag` Struct Reference

Inheritance diagram for `__gnu_pbds::rc_binomial_heap_tag`:



5.378.1 Detailed Description

Redundant-counter binomial-heap.

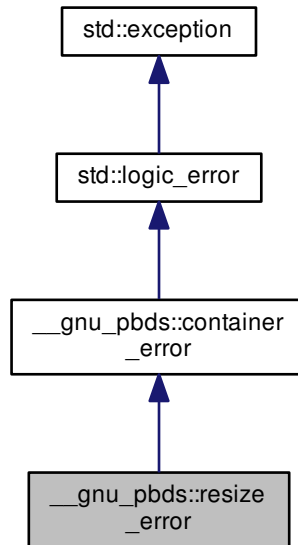
Definition at line 180 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.379 `__gnu_pbds::resize_error` Struct Reference

Inheritance diagram for `__gnu_pbds::resize_error`:



Public Member Functions

- virtual const char * [what](#) () const `_GLIBCXX_TXN_SAFE_DYN` noexcept

5.379.1 Detailed Description

A container cannot be resized.

Definition at line 73 of file `exception.hpp`.

5.379.2 Member Function Documentation

5.379.2.1 virtual const char* `std::logic_error::what` () const `[virtual]`, `[noexcept]`, `[inherited]`

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

Reimplemented in [std::future_error](#).

The documentation for this struct was generated from the following file:

- [exception.hpp](#)

5.380 `__gnu_pbds::sample_probe_fn` Class Reference

Public Types

- typedef `std::size_t` **size_type**

Public Member Functions

- [sample_probe_fn](#) ()
- [sample_probe_fn](#) (const [sample_probe_fn](#) &)
- void [swap](#) ([sample_probe_fn](#) &)

Protected Member Functions

- `size_type` [operator\(\)](#) (key_const_reference *r_key*, `size_type` *i*) const

5.380.1 Detailed Description

A sample probe policy.

Definition at line 47 of file `sample_probe_fn.hpp`.

5.380.2 Constructor & Destructor Documentation

5.380.2.1 `__gnu_pbds::sample_probe_fn::sample_probe_fn ()`

Default constructor.

5.380.2.2 `__gnu_pbds::sample_probe_fn::sample_probe_fn (const sample_probe_fn &)`

Copy constructor.

5.380.3 Member Function Documentation

5.380.3.1 `size_type __gnu_pbds::sample_probe_fn::operator() (key_const_reference r_key, size_type i) const` `[inline]`, `[protected]`

Returns the *i*-th offset from the hash value of some key *r_key*.

5.380.3.2 `void __gnu_pbds::sample_probe_fn::swap (sample_probe_fn &)` `[inline]`

Swaps content.

The documentation for this class was generated from the following file:

- [sample_probe_fn.hpp](#)

5.381 `__gnu_pbds::sample_range_hashing` Class Reference

Public Types

- typedef std::size_t [size_type](#)

Public Member Functions

- [sample_range_hashing](#) ()
- [sample_range_hashing](#) (const [sample_range_hashing](#) &other)
- void [swap](#) ([sample_range_hashing](#) &other)

Protected Member Functions

- void [notify_resized](#) ([size_type](#))
- [size_type](#) operator() ([size_type](#)) const

5.381.1 Detailed Description

A sample range-hashing functor.

Definition at line 47 of file `sample_range_hashing.hpp`.

5.381.2 Member Typedef Documentation

5.381.2.1 typedef std::size_t `__gnu_pbds::sample_range_hashing::size_type`

Size type.

Definition at line 51 of file `sample_range_hashing.hpp`.

5.381.3 Constructor & Destructor Documentation

5.381.3.1 `__gnu_pbds::sample_range_hashing::sample_range_hashing ()`

Default constructor.

5.381.3.2 `__gnu_pbds::sample_range_hashing::sample_range_hashing (const sample_range_hashing & other)`

Copy constructor.

5.381.4 Member Function Documentation

5.381.4.1 `void __gnu_pbds::sample_range_hashing::notify_resized (size_type)` `[protected]`

Notifies the policy object that the container's size has changed to argument's size.

5.381.4.2 `size_type __gnu_pbds::sample_range_hashing::operator() (size_type) const` `[inline]`, `[protected]`

Transforms the `__hash` value `hash` into a ranged-hash value.

5.381.4.3 `void __gnu_pbds::sample_range_hashing::swap (sample_range_hashing & other)` `[inline]`

Swaps content.

The documentation for this class was generated from the following file:

- [sample_range_hashing.hpp](#)

5.382 `__gnu_pbds::sample_ranged_hash_fn` Class Reference

Public Types

- typedef `std::size_t` **size_type**

Public Member Functions

- [sample_ranged_hash_fn](#) ()
- [sample_ranged_hash_fn](#) (const [sample_ranged_hash_fn](#) &)
- void [swap](#) ([sample_ranged_hash_fn](#) &)

Protected Member Functions

- void [notify_resized](#) (size_type)
- size_type [operator\(\)](#) (key_const_reference) const

5.382.1 Detailed Description

A sample ranged-hash functor.

Definition at line 47 of file `sample_ranged_hash_fn.hpp`.

5.382.2 Constructor & Destructor Documentation

5.382.2.1 `__gnu_pbds::sample_ranged_hash_fn::sample_ranged_hash_fn ()`

Default constructor.

5.382.2.2 `__gnu_pbds::sample_ranged_hash_fn::sample_ranged_hash_fn (const sample_ranged_hash_fn &)`

Copy constructor.

5.382.3 Member Function Documentation

5.382.3.1 `void __gnu_pbds::sample_ranged_hash_fn::notify_resized (size_type)` `[protected]`

Notifies the policy object that the container's `__size` has changed to `size`.

5.382.3.2 `size_type __gnu_pbds::sample_ranged_hash_fn::operator() (key_const_reference) const` `[inline]`, `[protected]`

Transforms `key_const_reference` into a position within the table.

5.382.3.3 `void __gnu_pbds::sample_ranged_hash_fn::swap (sample_ranged_hash_fn &)` `[inline]`

Swaps content.

The documentation for this class was generated from the following file:

- [sample_ranged_hash_fn.hpp](#)

5.383 `__gnu_pbds::sample_ranged_probe_fn` Class Reference

Public Types

- `typedef std::size_t size_type`

Public Member Functions

- `sample_ranged_probe_fn` (const [sample_ranged_probe_fn](#) &)
- `void swap` ([sample_ranged_probe_fn](#) &)

Protected Member Functions

- `void notify_resized` (size_type)
- `size_type operator()` (key_const_reference, std::size_t, size_type) const

5.383.1 Detailed Description

A sample ranged-probe functor.

Definition at line 47 of file `sample_ranged_probe_fn.hpp`.

The documentation for this class was generated from the following file:

- [sample_ranged_probe_fn.hpp](#)

5.384 `__gnu_pbds::sample_resize_policy` Class Reference

Public Types

- typedef std::size_t [size_type](#)

Public Member Functions

- [sample_resize_policy](#) ()
- [sample_range_hashing](#) (const [sample_resize_policy](#) &other)
- void [swap](#) ([sample_resize_policy](#) &other)

Protected Member Functions

- [size_type](#) [get_new_size](#) ([size_type](#) size, [size_type](#) num_used_e) const
- bool [is_resize_needed](#) () const
- void [notify_cleared](#) ()
- void [notify_erase_search_collision](#) ()
- void [notify_erase_search_end](#) ()
- void [notify_erase_search_start](#) ()
- void [notify_erased](#) ([size_type](#) num_e)
- void [notify_find_search_collision](#) ()
- void [notify_find_search_end](#) ()
- void [notify_find_search_start](#) ()
- void [notify_insert_search_collision](#) ()
- void [notify_insert_search_end](#) ()
- void [notify_insert_search_start](#) ()
- void [notify_inserted](#) ([size_type](#) num_e)
- void [notify_resized](#) ([size_type](#) new_size)

5.384.1 Detailed Description

A sample resize policy.

Definition at line 47 of file `sample_resize_policy.hpp`.

5.384.2 Member Typedef Documentation

5.384.2.1 typedef std::size_t __gnu_pbds::sample_resize_policy::size_type

Size type.

Definition at line 51 of file sample_resize_policy.hpp.

5.384.3 Constructor & Destructor Documentation

5.384.3.1 __gnu_pbds::sample_resize_policy::sample_resize_policy ()

Default constructor.

5.384.4 Member Function Documentation

5.384.4.1 size_type __gnu_pbds::sample_resize_policy::get_new_size (size_type size, size_type num_used_e) const [protected]

Queries what the new size should be.

5.384.4.2 bool __gnu_pbds::sample_resize_policy::is_resize_needed () const [inline], [protected]

Queries whether a resize is needed.

5.384.4.3 void __gnu_pbds::sample_resize_policy::notify_cleared () [protected]

Notifies the table was cleared.

5.384.4.4 void __gnu_pbds::sample_resize_policy::notify_erase_search_collision () [inline], [protected]

Notifies a search encountered a collision.

5.384.4.5 void __gnu_pbds::sample_resize_policy::notify_erase_search_end () [inline], [protected]

Notifies a search ended.

5.384.4.6 void __gnu_pbds::sample_resize_policy::notify_erase_search_start () [inline], [protected]

Notifies a search started.

5.384.4.7 void __gnu_pbds::sample_resize_policy::notify_erased (size_type num_e) [inline], [protected]

Notifies an element was erased.

5.384.4.8 void __gnu_pbds::sample_resize_policy::notify_find_search_collision () [inline], [protected]

Notifies a search encountered a collision.

5.384.4.9 void __gnu_pbds::sample_resize_policy::notify_find_search_end () [inline], [protected]

Notifies a search ended.

5.384.4.10 void __gnu_pbds::sample_resize_policy::notify_find_search_start () [inline], [protected]

Notifies a search started.

5.384.4.11 void __gnu_pbds::sample_resize_policy::notify_insert_search_collision () [inline], [protected]

Notifies a search encountered a collision.

5.384.4.12 void __gnu_pbds::sample_resize_policy::notify_insert_search_end () [inline], [protected]

Notifies a search ended.

5.384.4.13 void __gnu_pbds::sample_resize_policy::notify_insert_search_start () [inline], [protected]

Notifies a search started.

5.384.4.14 void __gnu_pbds::sample_resize_policy::notify_inserted (size_type *num_e*) [inline], [protected]

Notifies an element was inserted.

5.384.4.15 void __gnu_pbds::sample_resize_policy::notify_resized (size_type *new_size*) [protected]

Notifies the table was resized to *new_size*.

5.384.4.16 __gnu_pbds::sample_resize_policy::sample_range_hashing (const sample_resize_policy & *other*)

Copy constructor.

5.384.4.17 void __gnu_pbds::sample_resize_policy::swap (sample_resize_policy & *other*) [inline]

Swaps content.

The documentation for this class was generated from the following file:

- [sample_resize_policy.hpp](#)

5.385 `__gnu_pbds::sample_resize_trigger` Class Reference

Public Types

- typedef std::size_t [size_type](#)

Public Member Functions

- [sample_resize_trigger](#) ()
- [sample_range_hashing](#) (const [sample_resize_trigger](#) &)
- void [swap](#) ([sample_resize_trigger](#) &)

Protected Member Functions

- bool [is_grow_needed](#) ([size_type](#) size, [size_type](#) num_entries) const
- bool [is_resize_needed](#) () const
- void [notify_cleared](#) ()
- void [notify_erase_search_collision](#) ()
- void [notify_erase_search_end](#) ()
- void [notify_erase_search_start](#) ()
- void [notify_erased](#) ([size_type](#) num_entries)
- void [notify_externally_resized](#) ([size_type](#) new_size)
- void [notify_find_search_collision](#) ()
- void [notify_find_search_end](#) ()
- void [notify_find_search_start](#) ()
- void [notify_insert_search_collision](#) ()
- void [notify_insert_search_end](#) ()
- void [notify_insert_search_start](#) ()
- void [notify_inserted](#) ([size_type](#) num_entries)
- void [notify_resized](#) ([size_type](#) new_size)

5.385.1 Detailed Description

A sample resize trigger policy.

Definition at line 47 of file `sample_resize_trigger.hpp`.

5.385.2 Member Typedef Documentation

5.385.2.1 typedef std::size_t `__gnu_pbds::sample_resize_trigger::size_type`

Size type.

Definition at line 51 of file `sample_resize_trigger.hpp`.

5.385.3 Constructor & Destructor Documentation

5.385.3.1 __gnu_pbds::sample_resize_trigger ()

Default constructor.

5.385.4 Member Function Documentation

5.385.4.1 bool __gnu_pbds::sample_resize_trigger::is_grow_needed (size_type size, size_type num_entries) const [inline], [protected]

Queries whether a grow is needed.

5.385.4.2 bool __gnu_pbds::sample_resize_trigger::is_resize_needed () const [inline], [protected]

Queries whether a resize is needed.

5.385.4.3 void __gnu_pbds::sample_resize_trigger::notify_cleared () [protected]

Notifies the table was cleared.

5.385.4.4 void __gnu_pbds::sample_resize_trigger::notify_erase_search_collision () [inline], [protected]

Notifies a search encountered a collision.

5.385.4.5 void __gnu_pbds::sample_resize_trigger::notify_erase_search_end () [inline], [protected]

Notifies a search ended.

5.385.4.6 void __gnu_pbds::sample_resize_trigger::notify_erase_search_start () [inline], [protected]

Notifies a search started.

5.385.4.7 void __gnu_pbds::sample_resize_trigger::notify_erased (size_type num_entries) [inline], [protected]

Notifies an element was erased.

5.385.4.8 void __gnu_pbds::sample_resize_trigger::notify_externally_resized (size_type new_size) [protected]

Notifies the table was resized externally.

5.385.4.9 void __gnu_pbds::sample_resize_trigger::notify_find_search_collision () [inline], [protected]

Notifies a search encountered a collision.

5.385.4.10 `void __gnu_pbds::sample_resize_trigger::notify_find_search_end () [inline], [protected]`

Notifies a search ended.

5.385.4.11 `void __gnu_pbds::sample_resize_trigger::notify_find_search_start () [inline], [protected]`

Notifies a search started.

5.385.4.12 `void __gnu_pbds::sample_resize_trigger::notify_insert_search_collision () [inline], [protected]`

Notifies a search encountered a collision.

5.385.4.13 `void __gnu_pbds::sample_resize_trigger::notify_insert_search_end () [inline], [protected]`

Notifies a search ended.

5.385.4.14 `void __gnu_pbds::sample_resize_trigger::notify_insert_search_start () [inline], [protected]`

Notifies a search started.

5.385.4.15 `void __gnu_pbds::sample_resize_trigger::notify_inserted (size_type num_entries) [inline], [protected]`

Notifies an element was inserted. the total number of entries in the table is `num_entries`.

5.385.4.16 `void __gnu_pbds::sample_resize_trigger::notify_resized (size_type new_size) [protected]`

Notifies the table was resized as a result of this object's signifying that a resize is needed.

5.385.4.17 `__gnu_pbds::sample_resize_trigger::sample_range_hashing (const sample_resize_trigger &)`

Copy constructor.

5.385.4.18 `void __gnu_pbds::sample_resize_trigger::swap (sample_resize_trigger &) [inline]`

Swaps content.

The documentation for this class was generated from the following file:

- [sample_resize_trigger.hpp](#)

5.386 `__gnu_pbds::sample_size_policy` Class Reference

Public Types

- `typedef std::size_t size_type`

Public Member Functions

- [sample_size_policy](#) ()
- [sample_range_hashing](#) (const [sample_size_policy](#) &)
- void [swap](#) ([sample_size_policy](#) &other)

Protected Member Functions

- [size_type get_nearest_larger_size](#) ([size_type](#) size) const
- [size_type get_nearest_smaller_size](#) ([size_type](#) size) const

5.386.1 Detailed Description

A sample size policy.

Definition at line 47 of file `sample_size_policy.hpp`.

5.386.2 Member Typedef Documentation

5.386.2.1 `typedef std::size_t __gnu_pbds::sample_size_policy::size_type`

Size type.

Definition at line 51 of file `sample_size_policy.hpp`.

5.386.3 Constructor & Destructor Documentation

5.386.3.1 `__gnu_pbds::sample_size_policy::sample_size_policy ()`

Default constructor.

5.386.4 Member Function Documentation

5.386.4.1 `size_type __gnu_pbds::sample_size_policy::get_nearest_larger_size (size_type size) const` [inline], [protected]

Given a `__size` size, returns a `__size` that is larger.

5.386.4.2 `size_type __gnu_pbds::sample_size_policy::get_nearest_smaller_size (size_type size) const` [inline], [protected]

Given a `__size` size, returns a `__size` that is smaller.

5.386.4.3 `__gnu_pbds::sample_size_policy::sample_range_hashing (const sample_size_policy &)`

Copy constructor.

5.386.4.4 `void __gnu_pbds::sample_size_policy::swap (sample_size_policy & other) [inline]`

Swaps content.

The documentation for this class was generated from the following file:

- [sample_size_policy.hpp](#)

5.387 `__gnu_pbds::sample_tree_node_update< Const_Node_Iter, Node_Iter, Cmp_Fn, _Alloc >` Class Template Reference

5.387.1 Detailed Description

```
template<typename Const_Node_Iter, typename Node_Iter, typename Cmp_Fn, typename _Alloc>
class __gnu_pbds::sample_tree_node_update< Const_Node_Iter, Node_Iter, Cmp_Fn, _Alloc >
```

A sample node updatator.

Definition at line 49 of file `sample_tree_node_update.hpp`.

The documentation for this class was generated from the following file:

- [sample_tree_node_update.hpp](#)

5.388 `__gnu_pbds::sample_trie_access_traits` Struct Reference

Public Types

- enum { **max_size** }
- typedef `_Alloc::template rebind< key_type > __rebind_k`
- typedef `std::string::const_iterator` **const_iterator**
- typedef char [e_type](#)
- typedef `__rebind_k::other::const_reference` **key_const_reference**
- typedef [std::string](#) **key_type**
- typedef `std::size_t` **size_type**

Static Public Member Functions

- static `const_iterator` [begin](#) (`key_const_reference`)
- static `size_type` [e_pos](#) ([e_type](#))
- static `const_iterator` [end](#) (`key_const_reference`)

5.388.1 Detailed Description

A sample trie element access traits.

Definition at line 47 of file `sample_trie_access_traits.hpp`.

5.388.2 Member Typedef Documentation

5.388.2.1 typedef char __gnu_pbds::sample_trie_access_traits::e_type

Element type.

Definition at line 57 of file `sample_trie_access_traits.hpp`.

5.388.3 Member Function Documentation

5.388.3.1 static const_iterator __gnu_pbds::sample_trie_access_traits::begin (key_const_reference) [inline], [static]

Returns a `const_iterator` to the first element of `r_key`.

5.388.3.2 static size_type __gnu_pbds::sample_trie_access_traits::e_pos (e_type) [inline], [static]

Maps an element to a position.

5.388.3.3 static const_iterator __gnu_pbds::sample_trie_access_traits::end (key_const_reference) [inline], [static]

Returns a `const_iterator` to the after-last element of `r_key`.

The documentation for this struct was generated from the following file:

- [sample_trie_access_traits.hpp](#)

5.389 **__gnu_pbds::sample_trie_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc > Class Template Reference**

Public Types

- typedef std::size_t **metadata_type**

Protected Member Functions

- [sample_trie_node_update](#) ()
- void [operator\(\)](#) (node_iterator, node_const_iterator) const

5.389.1 Detailed Description

```
template<typename Node_Cltr, typename Node_Itr, typename _ATraits, typename _Alloc>
class __gnu_pbds::sample_trie_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >
```

A sample node updatator.

Definition at line 49 of file sample_trie_node_update.hpp.

5.389.2 Constructor & Destructor Documentation

```
5.389.2.1 template<typename Node_Cltr , typename Node_Itr , typename _ATraits , typename _Alloc >
        __gnu_pbds::sample_trie_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::sample_trie_node_update
        ( ) [protected]
```

Default constructor.

5.389.3 Member Function Documentation

```
5.389.3.1 template<typename Node_Cltr , typename Node_Itr , typename _ATraits , typename _Alloc > void
        __gnu_pbds::sample_trie_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::operator() ( node_iterator ,
        node_const_iterator ) const [inline], [protected]
```

Updates the rank of a node through a node_iterator node_it; end_nd_it is the end node iterator.

The documentation for this class was generated from the following file:

- [sample_trie_node_update.hpp](#)

5.390 __gnu_pbds::sample_update_policy Struct Reference

Public Member Functions

- [sample_update_policy](#) ()
- [sample_update_policy](#) (const [sample_update_policy](#) &)
- void [swap](#) ([sample_update_policy](#) &other)

Protected Types

- typedef some_metadata_type [metadata_type](#)

Protected Member Functions

- [metadata_type](#) [operator\(\)](#) () const
- bool [operator\(\)](#) (metadata_reference) const

5.390.1 Detailed Description

A sample list-update policy.

Definition at line 47 of file `sample_update_policy.hpp`.

5.390.2 Member Typedef Documentation

5.390.2.1 `typedef some_metadata_type __gnu_pbds::sample_update_policy::metadata_type` `[protected]`

Metadata on which this functor operates.

Definition at line 61 of file `sample_update_policy.hpp`.

5.390.3 Constructor & Destructor Documentation

5.390.3.1 `__gnu_pbds::sample_update_policy::sample_update_policy ()`

Default constructor.

5.390.3.2 `__gnu_pbds::sample_update_policy::sample_update_policy (const sample_update_policy &)`

Copy constructor.

5.390.4 Member Function Documentation

5.390.4.1 `metadata_type __gnu_pbds::sample_update_policy::operator() () const` `[protected]`

Creates a metadata object.

5.390.4.2 `bool __gnu_pbds::sample_update_policy::operator() (metadata_reference) const` `[protected]`

Decides whether a metadata object should be moved to the front of the list. A list-update based containers object will call this method to decide whether to move a node to the front of the list. The method should return true if the node should be moved to the front of the list.

5.390.4.3 `void __gnu_pbds::sample_update_policy::swap (sample_update_policy & other)` `[inline]`

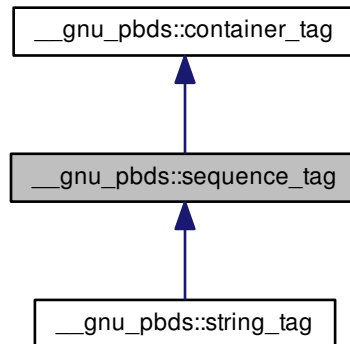
Swaps content.

The documentation for this struct was generated from the following file:

- [sample_update_policy.hpp](#)

5.391 `__gnu_pbds::sequence_tag` Struct Reference

Inheritance diagram for `__gnu_pbds::sequence_tag`:



5.391.1 Detailed Description

Basic sequence.

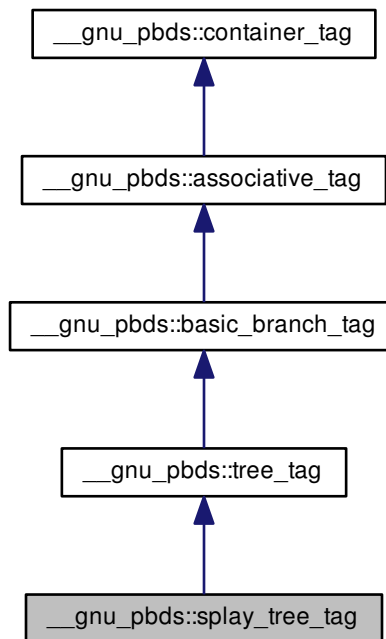
Definition at line 129 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.392 __gnu_pbds::splay_tree_tag Struct Reference

Inheritance diagram for __gnu_pbds::splay_tree_tag:



5.392.1 Detailed Description

Splay tree.

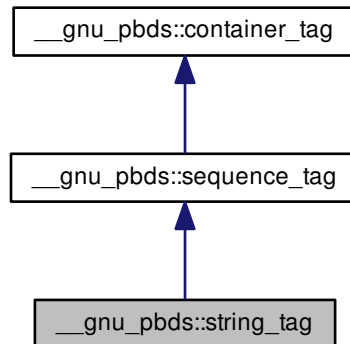
Definition at line 156 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.393 `__gnu_pbds::string_tag` Struct Reference

Inheritance diagram for `__gnu_pbds::string_tag`:



5.393.1 Detailed Description

Basic string container, inclusive of strings, ropes, etc.

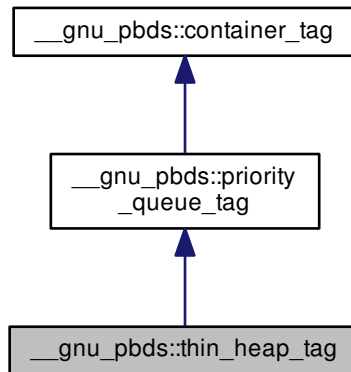
Definition at line 132 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.394 __gnu_pbds::thin_heap_tag Struct Reference

Inheritance diagram for __gnu_pbds::thin_heap_tag:



5.394.1 Detailed Description

Thin heap.

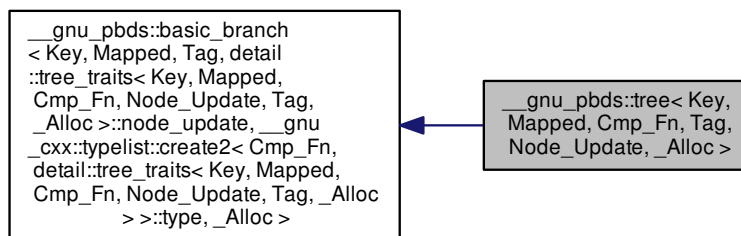
Definition at line 186 of file tag_and_trait.hpp.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.395 __gnu_pbds::tree< Key, Mapped, Cmp_Fn, Tag, Node_Update, _Alloc > Class Template Reference

Inheritance diagram for __gnu_pbds::tree< Key, Mapped, Cmp_Fn, Tag, Node_Update, _Alloc >:



Public Types

- typedef Cmp_Fn [cmp_fn](#)
- typedef [detail::tree_traits](#)< Key, Mapped, Cmp_Fn, Node_Update, Tag, _Alloc >::node_update **node_update**

Public Member Functions

- [tree](#) (const [cmp_fn](#) &c)
- template<typename It >
[tree](#) (It first, It last)
- template<typename It >
[tree](#) (It first, It last, const [cmp_fn](#) &c)
- **tree** (const [tree](#) &other)
- [tree](#) & **operator=** (const [tree](#) &other)
- void **swap** ([tree](#) &other)

5.395.1 Detailed Description

```
template<typename Key, typename Mapped, typename Cmp_Fn = std::less<Key>, typename Tag = rb_tree_tag, template< typename
Node_Cltr, typename Node_Itr, typename Cmp_Fn_, typename _Alloc_ > class Node_Update = null_node_update, typename _Alloc =
std::allocator<char>>>
class __gnu_pbds::tree< Key, Mapped, Cmp_Fn, Tag, Node_Update, _Alloc >
```

A tree-based container.

Template Parameters

<i>Key</i>	Key type.
<i>Mapped</i>	Map type.
<i>Cmp_Fn</i>	Comparison functor.
<i>Tag</i>	Instantiating data structure type, see container_tag.
<i>Node_Update</i>	Updates tree internal-nodes, restores invariants when invalidated. XXX See design::tree-based-containersnode invariants.
<i>_Alloc</i>	Allocator type.

Base tag choices are: [ov_tree_tag](#), [rb_tree_tag](#), [splay_tree_tag](#).

Base is [basic_branch](#).

Definition at line 635 of file [assoc_container.hpp](#).

5.395.2 Member Typedef Documentation

```
5.395.2.1  template<typename Key , typename Mapped , typename Cmp_Fn = std::less<Key>, typename Tag = rb_tree_tag,
template< typename Node_Cltr, typename Node_Itr, typename Cmp_Fn_, typename _Alloc_ > class Node_Update =
null_node_update, typename _Alloc = std::allocator<char>>> typedef Cmp_Fn __gnu_pbds::tree< Key, Mapped,
Cmp_Fn, Tag, Node_Update, _Alloc >::cmp_fn
```

Comparison functor type.

Definition at line 642 of file `assoc_container.hpp`.

5.395.3 Constructor & Destructor Documentation

```
5.395.3.1  template<typename Key , typename Mapped , typename Cmp_Fn = std::less<Key>, typename Tag = rb_tree_tag,
template< typename Node_Cltr, typename Node_Itr, typename Cmp_Fn_, typename _Alloc_ > class Node_Update =
null_node_update, typename _Alloc = std::allocator<char>>> __gnu_pbds::tree< Key, Mapped, Cmp_Fn, Tag,
Node_Update, _Alloc >::tree ( const cmp_fn & c ) [inline]
```

Constructor taking some policy objects. `r_cmp_fn` will be copied by the `Cmp_Fn` object of the container object.

Definition at line 648 of file `assoc_container.hpp`.

```
5.395.3.2  template<typename Key , typename Mapped , typename Cmp_Fn = std::less<Key>, typename Tag = rb_tree_tag,
template< typename Node_Cltr, typename Node_Itr, typename Cmp_Fn_, typename _Alloc_ > class Node_Update =
null_node_update, typename _Alloc = std::allocator<char>>> template<typename It > __gnu_pbds::tree< Key,
Mapped, Cmp_Fn, Tag, Node_Update, _Alloc >::tree ( It first, It last ) [inline]
```

Constructor taking `__iterators` to a range of `value_types`. The `value_types` between `first_it` and `last_it` will be inserted into the container object.

Definition at line 655 of file `assoc_container.hpp`.

```
5.395.3.3  template<typename Key , typename Mapped , typename Cmp_Fn = std::less<Key>, typename Tag = rb_tree_tag,
template< typename Node_Cltr, typename Node_Itr, typename Cmp_Fn_, typename _Alloc_ > class Node_Update =
null_node_update, typename _Alloc = std::allocator<char>>> template<typename It > __gnu_pbds::tree< Key,
Mapped, Cmp_Fn, Tag, Node_Update, _Alloc >::tree ( It first, It last, const cmp_fn & c ) [inline]
```

Constructor taking `__iterators` to a range of `value_types` and some policy objects The `value_types` between `first_it` and `last_it` will be inserted into the container object. `r_cmp_fn` will be copied by the `cmp_fn` object of the container object.

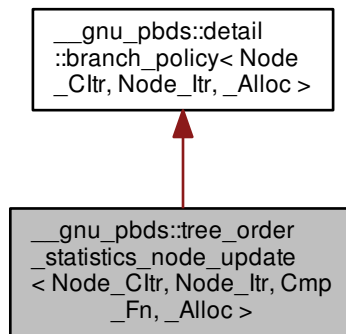
Definition at line 663 of file `assoc_container.hpp`.

The documentation for this class was generated from the following file:

- [assoc_container.hpp](#)

5.396 `__gnu_pbds::tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc >` Class Template Reference

Inheritance diagram for `__gnu_pbds::tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc >`:



Public Types

- typedef `_Alloc` **allocator_type**
- typedef `Cmp_Fn` **cmp_fn**
- typedef `node_const_iterator::value_type` **const_iterator**
- typedef `node_iterator::value_type` **iterator**
- typedef `base_type::key_const_reference` **key_const_reference**
- typedef `base_type::key_type` **key_type**
- typedef `size_type` **metadata_type**
- typedef `Node_Cltr` **node_const_iterator**
- typedef `Node_Itr` **node_iterator**
- typedef `allocator_type::size_type` **size_type**

Public Member Functions

- `const_iterator` [find_by_order](#) (`size_type`) const
- `iterator` [find_by_order](#) (`size_type`)
- `size_type` [order_of_key](#) (`key_const_reference`) const

Protected Member Functions

- void [operator\(\)](#) (`node_iterator`, `node_const_iterator`) const

Private Types

- typedef Node_Itr::value_type **it_type**
- typedef remove_const< key_type >::type **rckey_type**
- typedef remove_const< value_type >::type **rcvalue_type**
- typedef _Alloc::template rebind< rckey_type >::other **rebind_k**
- typedef _Alloc::template rebind< rcvalue_type >::other **rebind_v**
- typedef rebind_v::reference **reference**
- typedef std::iterator_traits< it_type >::value_type **value_type**

Private Member Functions

- virtual it_type **end** ()=0
- it_type **end_iterator** () const

Static Private Member Functions

- static key_const_reference **extract_key** (const_reference r_val)

5.396.1 Detailed Description

```
template<typename Node_Cltr, typename Node_Itr, typename Cmp_Fn, typename _Alloc>
class __gnu_pbds::tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc >
```

Functor updating ranks of entrees.

Definition at line 64 of file tree_policy.hpp.

5.396.2 Member Function Documentation

```
5.396.2.1 template<typename Node_Cltr , typename Node_Itr , typename Cmp_Fn , typename _Alloc >
tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc >::const_iterator
__gnu_pbds::tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc >::find_by_order (
size_type order ) const [inline]
```

Finds an entry by `__order`. Returns a `const_iterator` to the entry with the `__order` order, or a `const_iterator` to the container object's end if order is at least the size of the container object.

Definition at line 72 of file tree_policy.hpp.

```
5.396.2.2 template<typename Node_Cltr , typename Node_Itr , typename Cmp_Fn , typename _Alloc >
tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc >::iterator
__gnu_pbds::tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc >::find_by_order (
size_type order ) [inline]
```

Finds an entry by `__order`. Returns an iterator to the entry with the `__order` order, or an iterator to the container object's end if order is at least the size of the container object.

Definition at line 45 of file tree_policy.hpp.


```
5.396.2.3  template<typename Node_Cltr , typename Node_Itr , typename Cmp_Fn , typename _Alloc > void
            __gnu_pbds::tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc >::operator() (
            node_iterator node_it, node_const_iterator end_nd_it ) const    [inline], [protected]
```

Updates the rank of a node through a node_iterator node_it; end_nd_it is the end node iterator.

Definition at line 108 of file tree_policy.hpp.

```
5.396.2.4  template<typename Node_Cltr , typename Node_Itr , typename Cmp_Fn , typename _Alloc >
            tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc >::size_type
            __gnu_pbds::tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc >::order_of_key (
            key_const_reference r_key ) const    [inline]
```

Returns the order of a key within a sequence. For exapmle, if r_key is the smallest key, this method will return 0; if r_key is a key between the smallest and next key, this method will return 1; if r_key is a key larger than the largest key, this method will return the size of r_c.

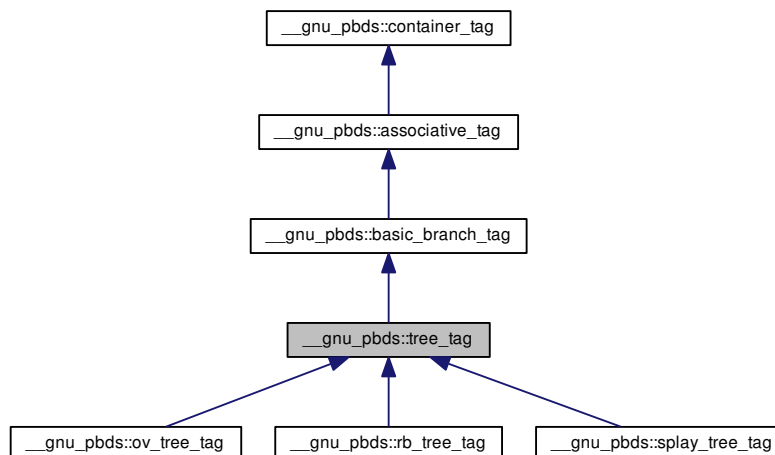
Definition at line 78 of file tree_policy.hpp.

The documentation for this class was generated from the following file:

- [tree_policy.hpp](#)

5.397 __gnu_pbds::tree_tag Struct Reference

Inheritance diagram for __gnu_pbds::tree_tag:



5.397.1 Detailed Description

Basic tree structure.

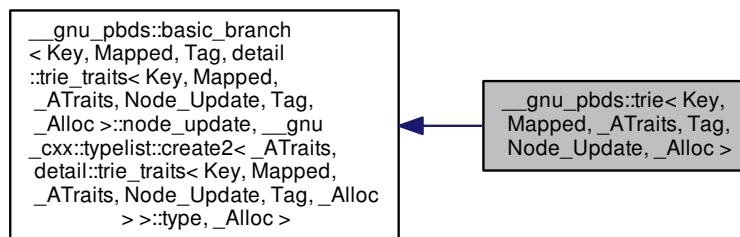
Definition at line 150 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.398 `__gnu_pbds::trie< Key, Mapped, _ATraits, Tag, Node_Update, _Alloc >` Class Template Reference

Inheritance diagram for `__gnu_pbds::trie< Key, Mapped, _ATraits, Tag, Node_Update, _Alloc >`:



Public Types

- typedef `_ATraits` [access_traits](#)
- typedef `detail::trie_traits< Key, Mapped, _ATraits, Node_Update, Tag, _Alloc >::node_update` **node_update**

Public Member Functions

- [trie](#) (const [access_traits](#) &t)
- template<typename It >
[trie](#) (It first, It last)
- template<typename It >
[trie](#) (It first, It last, const [access_traits](#) &t)
- **trie** (const [trie](#) &other)
- [trie](#) & **operator=** (const [trie](#) &other)
- void **swap** ([trie](#) &other)

5.398.1 Detailed Description

```

template<typename Key, typename Mapped, typename _ATraits = typename detail::default_trie_access_traits<Key>::type, typename
Tag = pat_trie_tag, template< typename Node_Citr, typename Node_Itr, typename _ATraits_, typename _Alloc_ > class Node_Update
= null_node_update, typename _Alloc = std::allocator<char>>>
class __gnu_pbds::trie< Key, Mapped, _ATraits, Tag, Node_Update, _Alloc >
  
```

A trie-based container.

Template Parameters

<i>Key</i>	Key type.
<i>Mapped</i>	Map type.
<i>_ATraits</i>	Element access traits.
<i>Tag</i>	Instantiating data structure type, see container_tag.
<i>Node_Update</i>	Updates trie internal-nodes, restores invariants when invalidated. XXX See design::tree-based-containersnode invariants.
<i>_Alloc</i>	Allocator type.

Base tag choice is pat_trie_tag.

Base is basic_branch.

Definition at line 731 of file assoc_container.hpp.

5.398.2 Member Typedef Documentation

5.398.2.1 `template<typename Key , typename Mapped , typename _ATraits = typename detail::default_trie_access_traits<↵
Key>::type, typename Tag = pat_trie_tag, template< typename Node_Cltr, typename Node_Itr, typename _ATraits_,
typename _Alloc_ > class Node_Update = null_node_update, typename _Alloc = std::allocator<char>>> typedef _ATraits
__gnu_pbds::trie< Key, Mapped, _ATraits, Tag, Node_Update, _Alloc >::access_traits`

Element access traits type.

Definition at line 738 of file assoc_container.hpp.

5.398.3 Constructor & Destructor Documentation

5.398.3.1 `template<typename Key , typename Mapped , typename _ATraits = typename detail::default_trie_access_traits<Key>↵
::type, typename Tag = pat_trie_tag, template< typename Node_Cltr, typename Node_Itr, typename _ATraits_, typename
Alloc > class Node_Update = null_node_update, typename _Alloc = std::allocator<char>>> __gnu_pbds::trie<
Key, Mapped, _ATraits, Tag, Node_Update, _Alloc >::trie (const access_traits & t) [inline]`

Constructor taking some policy objects. r_access_traits will be copied by the _ATraits object of the container object.

Definition at line 744 of file assoc_container.hpp.

5.398.3.2 `template<typename Key , typename Mapped , typename _ATraits = typename detail::default_trie_access_traits<Key>↵
::type, typename Tag = pat_trie_tag, template< typename Node_Cltr, typename Node_Itr, typename _ATraits_, typename
Alloc > class Node_Update = null_node_update, typename _Alloc = std::allocator<char>>> template<typename It >
__gnu_pbds::trie< Key, Mapped, _ATraits, Tag, Node_Update, _Alloc >::trie (It first, It last) [inline]`

Constructor taking __iterators to a range of value_types. The value_types between first_it and last_it will be inserted into the container object.

Definition at line 751 of file assoc_container.hpp.

```
5.398.3.3 template<typename Key , typename Mapped , typename _ATraits = typename detail::default_trie_access_traits<Key>↵
::type, typename Tag = pat_trie_tag, template< typename Node_Cltr, typename Node_Itr, typename _ATraits_, typename
_Alloc_ > class Node_Update = null_node_update, typename _Alloc = std::allocator<char>>> template<typename It >
__gnu_pbds::trie< Key, Mapped, _ATraits, Tag, Node_Update, _Alloc >::trie ( It first, It last, const access_traits &
t ) [inline]
```

Constructor taking `__iterators` to a range of `value_types` and some policy objects. The `value_types` between `first_it` and `last_it` will be inserted into the container object.

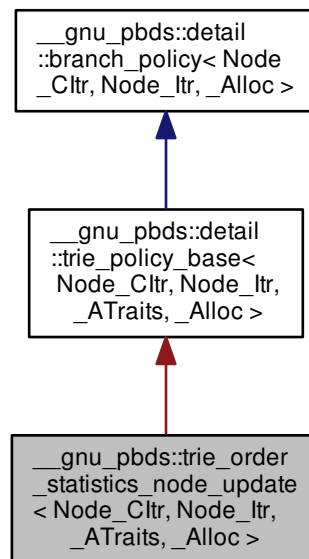
Definition at line 758 of file `assoc_container.hpp`.

The documentation for this class was generated from the following file:

- [assoc_container.hpp](#)

5.399 `__gnu_pbds::trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc > Class` Template Reference

Inheritance diagram for `__gnu_pbds::trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >`:



Public Types

- typedef access_traits::const_iterator **a_const_iterator**
- typedef _ATraits **access_traits**
- typedef _Alloc **allocator_type**
- typedef node_const_iterator::value_type **const_iterator**
- typedef node_iterator::value_type **iterator**
- typedef base_type::key_const_reference **key_const_reference**
- typedef base_type::key_type **key_type**
- typedef size_type **metadata_type**
- typedef Node_Cltr **node_const_iterator**
- typedef Node_Itr **node_iterator**
- typedef allocator_type::size_type **size_type**

Public Member Functions

- const_iterator [find_by_order](#) (size_type) const
- iterator [find_by_order](#) (size_type)
- size_type [order_of_key](#) (key_const_reference) const
- size_type [order_of_prefix](#) (a_const_iterator, a_const_iterator) const

Protected Member Functions

- void [operator\(\)](#) (node_iterator, node_const_iterator) const

Private Types

- typedef Node_Itr::value_type **it_type**
- typedef remove_const< key_type >::type **rkey_type**
- typedef remove_const< value_type >::type **rcvalue_type**
- typedef _Alloc::template rebind< rkey_type >::other **rebind_k**
- typedef _Alloc::template rebind< rcvalue_type >::other **rebind_v**
- typedef rebind_v::reference **reference**
- typedef std::iterator_traits< it_type >::value_type **value_type**

Private Member Functions

- virtual const_iterator **end** () const =0
- it_type **end_iterator** () const
- virtual const access_traits & **get_access_traits** () const =0

Static Private Member Functions

- static size_type **common_prefix_len** (node_iterator, e_const_iterator, e_const_iterator, const access_traits &)
- static key_const_reference **extract_key** (const_reference r_val)
- static iterator **leftmost_it** (node_iterator)
- static bool **less** (e_const_iterator, e_const_iterator, e_const_iterator, e_const_iterator, const access_traits &)
- static iterator **rightmost_it** (node_iterator)

5.399.1 Detailed Description

```
template<typename Node_Cltr, typename Node_Itr, typename _ATraits, typename _Alloc>
class __gnu_pbds::trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >
```

Functor updating ranks of entrees.

Definition at line 253 of file `trie_policy.hpp`.

5.399.2 Member Function Documentation

```
5.399.2.1 template<typename Node_Cltr , typename Node_Itr , typename _ATraits , typename _Alloc >
    trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::const_iterator
    __gnu_pbds::trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::find_by_order (
        size_type order ) const    [inline]
```

Finds an entry by `__order`. Returns a `const_iterator` to the entry with the `__order` order, or a `const_iterator` to the container object's end if order is at least the size of the container object.

Definition at line 79 of file `trie_policy.hpp`.

```
5.399.2.2 template<typename Node_Cltr , typename Node_Itr , typename _ATraits , typename _Alloc
    > trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::iterator
    __gnu_pbds::trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::find_by_order (
        size_type order )    [inline]
```

Finds an entry by `__order`. Returns an iterator to the entry with the `__order` order, or an iterator to the container object's end if order is at least the size of the container object.

Definition at line 45 of file `trie_policy.hpp`.

```
5.399.2.3 template<typename Node_Cltr , typename Node_Itr , typename _ATraits , typename _Alloc > void
    __gnu_pbds::trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::operator() (
        node_iterator nd_it, node_const_iterator ) const    [inline], [protected]
```

Updates the rank of a node through a `node_iterator` `node_it`; `end_nd_it` is the end node iterator.

Definition at line 152 of file `trie_policy.hpp`.

```
5.399.2.4 template<typename Node_Cltr , typename Node_Itr , typename _ATraits , typename _Alloc >
    trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::size_type
    __gnu_pbds::trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::order_of_key (
        key_const_reference r_key ) const    [inline]
```

Returns the order of a key within a sequence. For exapmle, if `r_key` is the smallest key, this method will return 0; if `r_key` is a key between the smallest and next key, this method will return 1; if `r_key` is a key larger than the largest key, this method will return the size of `r_c`.

Definition at line 85 of file `trie_policy.hpp`.

```

5.399.2.5 template<typename Node_Cltr , typename Node_Itr , typename _ATraits , typename _Alloc >
    trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::size_type
    __gnu_pbds::trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::order_of_prefix (
        a_const_iterator b, a_const_iterator e ) const    [inline]

```

Returns the order of a prefix within a sequence. For example, if [b, e] is the smallest prefix, this method will return 0; if r_key is a key between the smallest and next key, this method will return 1; if r_key is a key larger than the largest key, this method will return the size of r_c.

Definition at line 96 of file trie_policy.hpp.

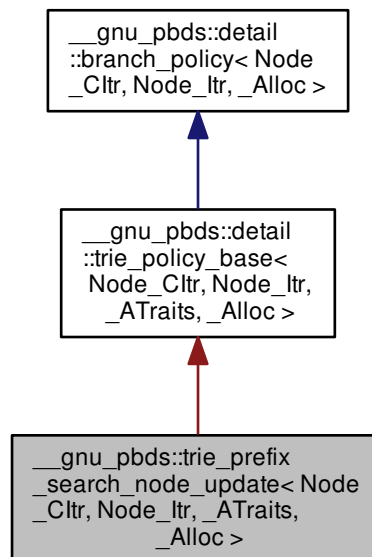
References `__gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc >::begin()`, `__gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc >::e_pos()`, and `__gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc >::end()`.

The documentation for this class was generated from the following file:

- [trie_policy.hpp](#)

5.400 `__gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >` Class Template Reference

Inheritance diagram for `__gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >`:



Public Types

- typedef `access_traits::const_iterator` [a_const_iterator](#)
- typedef `_ATraits` [access_traits](#)
- typedef `_Alloc` [allocator_type](#)
- typedef `node_const_iterator::value_type` **const_iterator**
- typedef `node_iterator::value_type` **iterator**
- typedef `base_type::key_const_reference` **key_const_reference**
- typedef `base_type::key_type` **key_type**
- typedef [null_type](#) **metadata_type**
- typedef `Node_Cltr` **node_const_iterator**
- typedef `Node_Itr` **node_iterator**
- typedef `allocator_type::size_type` [size_type](#)

Public Member Functions

- `std::pair< const_iterator, const_iterator >` [prefix_range](#) (`key_const_reference`) const
- `std::pair< iterator, iterator >` [prefix_range](#) (`key_const_reference`)
- `std::pair< const_iterator, const_iterator >` [prefix_range](#) ([a_const_iterator](#), [a_const_iterator](#)) const
- `std::pair< iterator, iterator >` [prefix_range](#) ([a_const_iterator](#), [a_const_iterator](#))

Protected Member Functions

- void [operator\(\)](#) (`node_iterator node_it`, `node_const_iterator end_nd_it`) const

Private Types

- typedef `rebind_v::const_pointer` **const_pointer**
- typedef `rebind_v::const_reference` **const_reference**
- typedef `Node_Itr::value_type` **it_type**
- typedef `remove_const< key_type >::type` **rckey_type**
- typedef `remove_const< value_type >::type` **rcvalue_type**
- typedef `_Alloc::template rebind< rckey_type >::other` **rebind_k**
- typedef `_Alloc::template rebind< rcvalue_type >::other` **rebind_v**
- typedef `rebind_v::reference` **reference**
- typedef `std::iterator_traits< it_type >::value_type` **value_type**

Private Member Functions

- `it_type` **end_iterator** () const

Static Private Member Functions

- static [size_type](#) **common_prefix_len** (`node_iterator`, `e_const_iterator`, `e_const_iterator`, const [access_traits](#) &)
- static `key_const_reference` **extract_key** (`const_reference r_val`)
- static `iterator` **leftmost_it** (`node_iterator`)
- static bool **less** (`e_const_iterator`, `e_const_iterator`, `e_const_iterator`, `e_const_iterator`, const [access_traits](#) &)
- static `iterator` **rightmost_it** (`node_iterator`)

5.400.1 Detailed Description

```
template<typename Node_Cltr, typename Node_Itr, typename _ATraits, typename _Alloc>
class __gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >
```

A node updatator that allows tries to be searched for the range of values that match a certain prefix.

Definition at line 155 of file trie_policy.hpp.

5.400.2 Member Typedef Documentation

```
5.400.2.1 template<typename Node_Cltr, typename Node_Itr, typename _ATraits, typename _Alloc> typedef
        access_traits::const_iterator __gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits,
        _Alloc >::a_const_iterator
```

Const element iterator.

Definition at line 168 of file trie_policy.hpp.

```
5.400.2.2 template<typename Node_Cltr, typename Node_Itr, typename _ATraits, typename _Alloc> typedef _ATraits
        __gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::access_traits
```

Element access traits.

Definition at line 165 of file trie_policy.hpp.

```
5.400.2.3 template<typename Node_Cltr, typename Node_Itr, typename _ATraits, typename _Alloc> typedef _Alloc
        __gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::allocator_type
```

_Alloc type.

Definition at line 171 of file trie_policy.hpp.

```
5.400.2.4 template<typename Node_Cltr, typename Node_Itr, typename _ATraits, typename _Alloc> typedef
        allocator_type::size_type __gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc
        >::size_type
```

Size type.

Definition at line 174 of file trie_policy.hpp.

5.400.3 Member Function Documentation

```
5.400.3.1 template<typename Node_Cltr , typename Node_Itr , typename _ATraits , typename _Alloc > void
        __gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::operator() (
        node_iterator node_it, node_const_iterator end_nd_it ) const [inline], [protected]
```

Called to update a node's metadata.

Definition at line 139 of file trie_policy.hpp.

5.400.3.2 `template<typename Node_Cltr , typename Node_Itr , typename _ATraits , typename _Alloc > std::pair<typename trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::const_iterator,typename trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::const_iterator >__gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::prefix_range (key_const_reference r_key) const`

Finds the const iterator range corresponding to all values whose prefixes match `r_key`.

Definition at line 47 of file `trie_policy.hpp`.

5.400.3.3 `template<typename Node_Cltr , typename Node_Itr , typename _ATraits , typename _Alloc > std::pair<typename trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::iterator,typename trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::iterator >__gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::prefix_range (key_const_reference r_key)`

Finds the iterator range corresponding to all values whose prefixes match `r_key`.

Definition at line 58 of file `trie_policy.hpp`.

5.400.3.4 `template<typename Node_Cltr , typename Node_Itr , typename _ATraits , typename _Alloc > std::pair<typename trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::const_iterator,typename trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::const_iterator >__gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::prefix_range (a_const_iterator b, a_const_iterator e) const`

Finds the const iterator range corresponding to all values whose prefixes match `[b, e)`.

Definition at line 69 of file `trie_policy.hpp`.

5.400.3.5 `template<typename Node_Cltr , typename Node_Itr , typename _ATraits , typename _Alloc > std::pair<typename trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::iterator,typename trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::iterator >__gnu_pbds::trie_prefix_search_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::prefix_range (a_const_iterator b, a_const_iterator e)`

Finds the iterator range corresponding to all values whose prefixes match `[b, e)`.

Definition at line 84 of file `trie_policy.hpp`.

The documentation for this class was generated from the following file:

- [trie_policy.hpp](#)

5.401 `__gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc >` Struct Template Reference

Public Types

- enum { **reverse** }
- enum { **min_e_val**, **max_e_val**, **max_size** }
- typedef `_Alloc::template rebind< key_type > __rebind_k`
- typedef `detail::__conditional_type< Reverse, typename String::const_reverse_iterator, typename String::const_iterator >::__type const_iterator`
- typedef `std::iterator_traits< const_iterator >::value_type e_type`
- typedef `__rebind_k::other::const_reference key_const_reference`
- typedef `String key_type`
- typedef `_Alloc::size_type size_type`

Static Public Member Functions

- static `const_iterator begin` (key_const_reference)
- static `size_type e_pos` (e_type e)
- static `const_iterator end` (key_const_reference)

5.401.1 Detailed Description

```
template<typename String = std::string, typename String::value_type Min_E_Val = detail::__numeric_traits<typename String::value_type>::__min,
typename String::value_type Max_E_Val = detail::__numeric_traits<typename String::value_type>::__max, bool Reverse = false,
typename _Alloc = std::allocator<char>>>
struct __gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc >
```

Element access traits for string types.

Template Parameters

<i>String</i>	String type.
<i>Min_E_Val</i>	Minimal element value.
<i>Max_E_Val</i>	Maximum element value.
<i>Reverse</i>	Reverse iteration should be used. Default: false.
<i>_Alloc</i>	Allocator type.

Definition at line 74 of file `trie_policy.hpp`.

5.401.2 Member Typedef Documentation

5.401.2.1 `template<typename String = std::string, typename String::value_type Min_E_Val = detail::__numeric_traits<typename String::value_type>::__min, typename String::value_type Max_E_Val = detail::__numeric_traits<typename String::value_type>::__max, bool Reverse = false, typename _Alloc = std::allocator<char>> typedef detail::__conditional_type<Reverse, typename String::const_reverse_iterator, typename String::const_iterator>::__type __gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc >::const_iterator`

Element const iterator type.

Definition at line 90 of file `trie_policy.hpp`.

5.401.2.2 `template<typename String = std::string, typename String::value_type Min_E_Val = detail::__numeric_traits<typename String::value_type>::__min, typename String::value_type Max_E_Val = detail::__numeric_traits<typename String::value_type>::__max, bool Reverse = false, typename _Alloc = std::allocator<char>> typedef std::iterator_traits<const_iterator>::value_type __gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc >::e_type`

Element type.

Definition at line 93 of file `trie_policy.hpp`.

5.401.3 Member Function Documentation

5.401.3.1 `template<typename String , typename String::value_type Min_E_Val, typename String::value_type Max_E_Val, bool Reverse, typename _Alloc > trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc >::const_iterator __gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc >::begin (key_const_reference r_key) [inline], [static]`

Returns a `const_iterator` to the first element of `key_const_reference` agumnet.

Definition at line 57 of file `trie_policy.hpp`.

Referenced by `__gnu_pbds::trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::order_of_← prefix()`.

5.401.3.2 `template<typename String , typename String::value_type Min_E_Val, typename String::value_type Max_E_Val, bool Reverse, typename _Alloc > trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc >::size_type __gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc >::e_pos (e_type e) [inline], [static]`

Maps an element to a position.

Definition at line 49 of file `trie_policy.hpp`.

Referenced by `__gnu_pbds::trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::order_of_← prefix()`.

```

5.401.3.3  template<typename String , typename String::value_type Min_E_Val, typename String::value_type Max_E_Val,
              bool Reverse, typename _Alloc > trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc
              >::const_iterator __gnu_pbds::trie_string_access_traits< String, Min_E_Val, Max_E_Val, Reverse, _Alloc
              >::end ( key_const_reference r_key )    [inline],[static]

```

Returns a const_iterator to the after-last element of key_const_reference argument.

Definition at line 65 of file trie_policy.hpp.

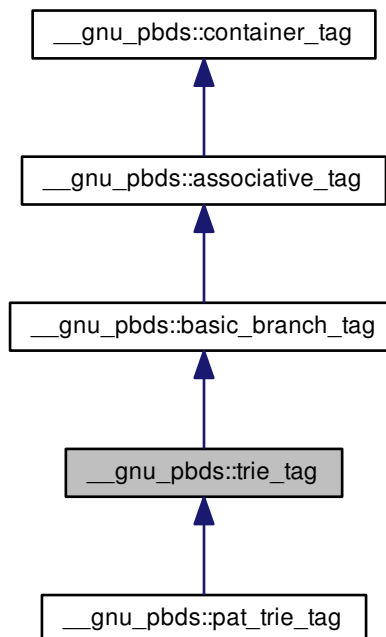
Referenced by __gnu_pbds::trie_order_statistics_node_update< Node_Cltr, Node_Itr, _ATraits, _Alloc >::order_of_↵
prefix().

The documentation for this struct was generated from the following file:

- [trie_policy.hpp](#)

5.402 __gnu_pbds::trie_tag Struct Reference

Inheritance diagram for __gnu_pbds::trie_tag:



5.402.1 Detailed Description

Basic trie structure.

Definition at line 162 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.403 `__gnu_pbds::trivial_iterator_tag` Struct Reference

5.403.1 Detailed Description

A trivial iterator tag. Signifies that the iterators has none of `std::iterators`'s movement abilities.

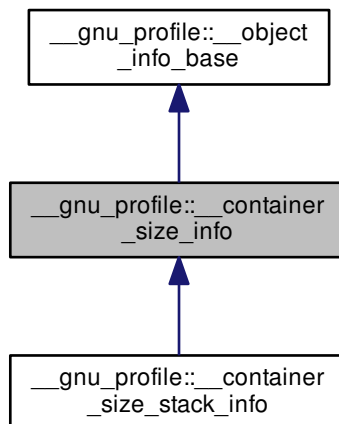
Definition at line 75 of file `tag_and_trait.hpp`.

The documentation for this struct was generated from the following file:

- [tag_and_trait.hpp](#)

5.404 `__gnu_profile::__container_size_info` Class Reference

Inheritance diagram for `__gnu_profile::__container_size_info`:



Public Member Functions

- **__container_size_info** (__stack_t __stack)
- [std::string __advice](#) () const
- void **__destruct** (std::size_t __num, std::size_t __inum)
- void **__init** (std::size_t __num)
- bool **__is_valid** () const
- float **__magnitude** () const
- void **__merge** (const [__container_size_info](#) &__o)
- void **__merge** (const [__object_info_base](#) &__o)
- void **__resize** (std::size_t __from, std::size_t __to)
- float **__resize_cost** (std::size_t __from, std::size_t)
- void **__set_invalid** ()
- __stack_t **__stack** () const
- void **__write** (FILE *__f) const

Protected Attributes

- __stack_t **_M_stack**
- bool **_M_valid**

5.404.1 Detailed Description

A container size instrumentation line in the object table.

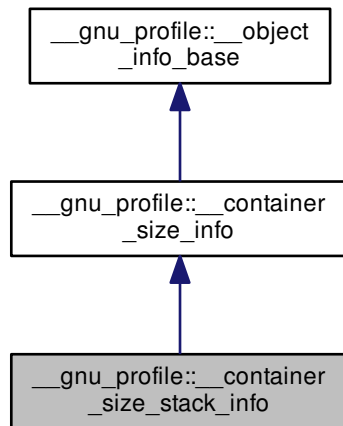
Definition at line 42 of file profiler_container_size.h.

The documentation for this class was generated from the following file:

- [profiler_container_size.h](#)

5.405 `__gnu_profile::__container_size_stack_info` Class Reference

Inheritance diagram for `__gnu_profile::__container_size_stack_info`:



Public Member Functions

- `__container_size_stack_info` (const `__container_size_info` &__o)
- `std::string __advice` () const
- void `__destruct` (std::size_t __num, std::size_t __inum)
- void `__init` (std::size_t __num)
- bool `__is_valid` () const
- float `__magnitude` () const
- void `__merge` (const `__container_size_info` &__o)
- void `__merge` (const `__object_info_base` &__o)
- void `__resize` (std::size_t __from, std::size_t __to)
- float `__resize_cost` (std::size_t __from, std::size_t)
- void `__set_invalid` ()
- `__stack_t __stack` () const
- void `__write` (FILE *__f) const

Protected Attributes

- `__stack_t __M_stack`
- bool `__M_valid`

5.405.1 Detailed Description

A container size instrumentation line in the stack table.

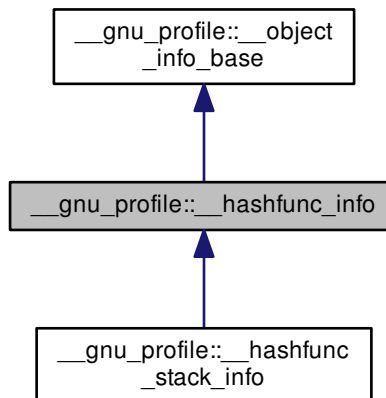
Definition at line 147 of file profiler_container_size.h.

The documentation for this class was generated from the following file:

- [profiler_container_size.h](#)

5.406 __gnu_profile::__hashfunc_info Class Reference

Inheritance diagram for __gnu_profile::__hashfunc_info:



Public Member Functions

- **__hashfunc_info** (__stack_t __stack)
- [std::string](#) **__advice** () const
- void **__destruct** (std::size_t __chain, std::size_t __accesses, std::size_t __hops)
- bool **__is_valid** () const
- float **__magnitude** () const
- void **__merge** (const [__hashfunc_info](#) &__o)
- void **__merge** (const [__object_info_base](#) &__o)
- void **__set_invalid** ()
- __stack_t **__stack** () const
- void **__write** (FILE *__f) const

Protected Attributes

- `__stack_t __M_stack`
- `bool __M_valid`

5.406.1 Detailed Description

A hash performance instrumentation line in the object table.

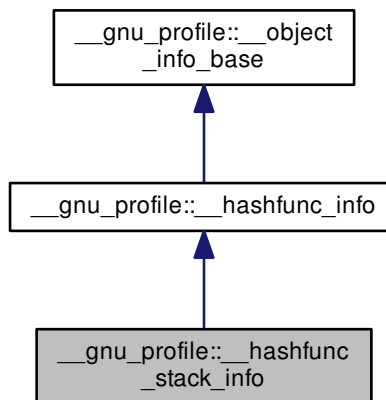
Definition at line 40 of file `profiler_hash_func.h`.

The documentation for this class was generated from the following file:

- [profiler_hash_func.h](#)

5.407 `__gnu_profile::__hashfunc_stack_info` Class Reference

Inheritance diagram for `__gnu_profile::__hashfunc_stack_info`:



Public Member Functions

- `__hashfunc_stack_info` (const [__hashfunc_info](#) &__o)
- `std::string __advice` () const
- `void __destruct` (std::size_t __chain, std::size_t __accesses, std::size_t __hops)
- `bool __is_valid` () const
- `float __magnitude` () const
- `void __merge` (const [__hashfunc_info](#) &__o)
- `void __merge` (const [__object_info_base](#) &__o)
- `void __set_invalid` ()
- `__stack_t __stack` () const
- `void __write` (FILE *__f) const

Protected Attributes

- `__stack_t __M_stack`
- `bool __M_valid`

5.407.1 Detailed Description

A hash performance instrumentation line in the stack table.

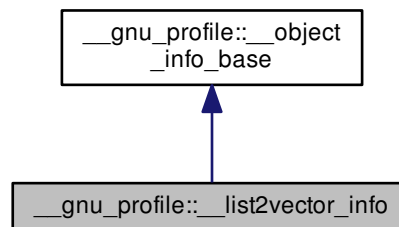
Definition at line 86 of file `profiler_hash_func.h`.

The documentation for this class was generated from the following file:

- [profiler_hash_func.h](#)

5.408 `__gnu_profile::__list2vector_info` Class Reference

Inheritance diagram for `__gnu_profile::__list2vector_info`:



Public Member Functions

- `__list2vector_info (__stack_t __stack)`
- `std::string __advice () const`
- `bool __is_valid () const`
- `std::size_t __iterate ()`
- `float __list_cost ()`
- `float __magnitude () const`
- `void __merge (const __list2vector_info &__o)`
- `void __merge (const __object_info_base &__o)`
- `void __opr_insert (std::size_t __shift, std::size_t __size)`
- `void __opr_iterate (int __num)`
- `std::size_t __resize ()`
- `void __resize (std::size_t __from, std::size_t)`
- `void __set_invalid ()`
- `void __set_list_cost (float __lc)`
- `void __set_vector_cost (float __vc)`
- `std::size_t __shift_count ()`
- `__stack_t __stack () const`
- `void __write (FILE *__f) const`

Protected Attributes

- `__stack_t __M_stack`
- `bool __M_valid`

5.408.1 Detailed Description

A list-to-vector instrumentation line in the object table.

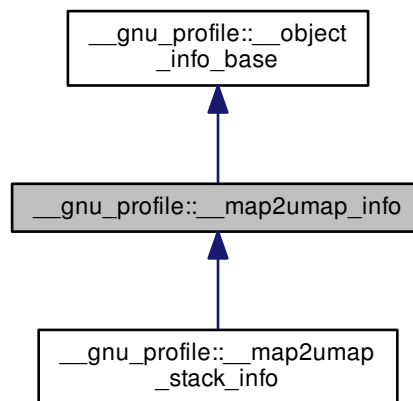
Definition at line 42 of file `profiler_list_to_vector.h`.

The documentation for this class was generated from the following file:

- [profiler_list_to_vector.h](#)

5.409 `__gnu_profile::__map2umap_info` Class Reference

Inheritance diagram for `__gnu_profile::__map2umap_info`:



Public Member Functions

- `__map2umap_info (__stack_t __stack)`
- `std::string __advice () const`
- `bool __is_valid () const`
- `float __magnitude () const`
- `void __merge (const __map2umap_info &__o)`
- `void __merge (const __object_info_base &__o)`

- void **__record_erase** (std::size_t __size, std::size_t __count)
- void **__record_find** (std::size_t __size)
- void **__record_insert** (std::size_t __size, std::size_t __count)
- void **__record_iterate** (int __count)
- void **__set_invalid** ()
- void **__set_iterate_costs** ()
- __stack_t **__stack** () const
- void **__write** (FILE *__f) const

Protected Attributes

- __stack_t **M_stack**
- bool **M_valid**

5.409.1 Detailed Description

A map-to-unordered_map instrumentation line in the object table.

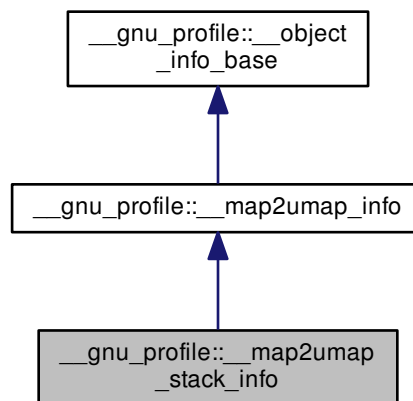
Definition at line 66 of file profiler_map_to_unordered_map.h.

The documentation for this class was generated from the following file:

- [profiler_map_to_unordered_map.h](#)

5.410 __gnu_profile::__map2umap_stack_info Class Reference

Inheritance diagram for __gnu_profile::__map2umap_stack_info:



Public Member Functions

- [__map2umap_stack_info](#) (const [__map2umap_info](#) &__o)
- [std::string __advice](#) () const
- [bool __is_valid](#) () const
- [float __magnitude](#) () const
- [void __merge](#) (const [__map2umap_info](#) &__o)
- [void __merge](#) (const [__object_info_base](#) &__o)
- [void __record_erase](#) (std::size_t __size, std::size_t __count)
- [void __record_find](#) (std::size_t __size)
- [void __record_insert](#) (std::size_t __size, std::size_t __count)
- [void __record_iterate](#) (int __count)
- [void __set_invalid](#) ()
- [void __set_iterate_costs](#) ()
- [__stack_t __stack](#) () const
- [void __write](#) (FILE * __f) const

Protected Attributes

- [__stack_t __M_stack](#)
- [bool __M_valid](#)

5.410.1 Detailed Description

A map-to-unordered_map instrumentation line in the stack table.

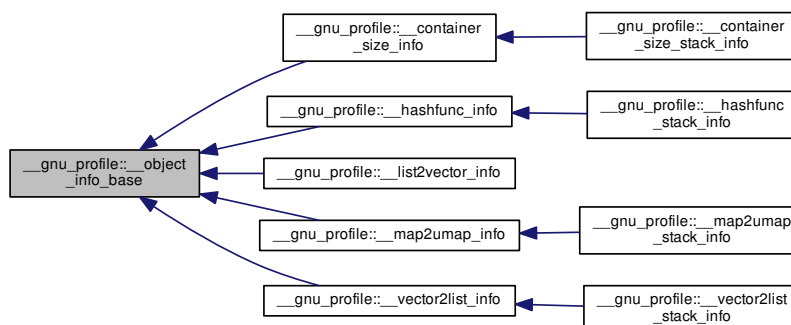
Definition at line 164 of file profiler_map_to_unordered_map.h.

The documentation for this class was generated from the following file:

- [profiler_map_to_unordered_map.h](#)

5.411 __gnu_profile::__object_info_base Class Reference

Inheritance diagram for __gnu_profile::__object_info_base:



Public Member Functions

- **__object_info_base** (__stack_t __stack)
- bool **__is_valid** () const
- void **__merge** (const [__object_info_base](#) &__o)
- void **__set_invalid** ()
- __stack_t **__stack** () const

Protected Attributes

- __stack_t **_M_stack**
- bool **_M_valid**

5.411.1 Detailed Description

Base class for a line in the object table.

Definition at line 127 of file profiler_node.h.

The documentation for this class was generated from the following file:

- [profiler_node.h](#)

5.412 __gnu_profile::__reentrance_guard Struct Reference

Static Public Member Functions

- static bool **__get_in** ()
- static bool & **__inside** ()

5.412.1 Detailed Description

Reentrance guard.

Mechanism to protect all __gnu_profile operations against recursion, multithreaded and exception reentrance.

Definition at line 58 of file profiler.h.

The documentation for this struct was generated from the following file:

- [profiler.h](#)

5.413 `__gnu_profile::__stack_hash` Class Reference

Public Member Functions

- `std::size_t operator() (__stack_t __s) const`
- `bool operator() (__stack_t __stack1, __stack_t __stack2) const`

5.413.1 Detailed Description

Hash function for summary trace using call stack as index.

Definition at line 93 of file `profiler_node.h`.

The documentation for this class was generated from the following file:

- [profiler_node.h](#)

5.414 `__gnu_profile::__trace_base< __object_info, __stack_info >` Class Template Reference

Public Member Functions

- `__object_info * __add_object (__stack_t __stack)`
- `void __collect_warnings (__warning_vector_t & __warnings)`
- `void __free ()`
- `void __retire_object (__object_info * __info)`
- `void __write (FILE * __f)`

Protected Attributes

- `const char * __id`

5.414.1 Detailed Description

```
template<typename __object_info, typename __stack_info>
class __gnu_profile::__trace_base< __object_info, __stack_info >
```

Base class for all trace producers.

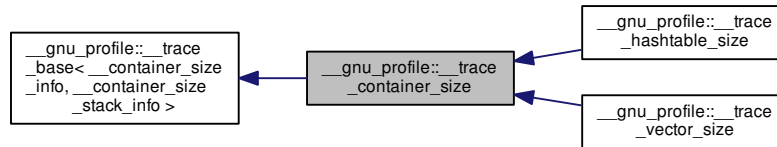
Definition at line 190 of file `profiler_trace.h`.

The documentation for this class was generated from the following file:

- [profiler_trace.h](#)

5.415 `__gnu_profile::__trace_container_size` Class Reference

Inheritance diagram for `__gnu_profile::__trace_container_size`:



Public Member Functions

- [__container_size_info](#) * **add_object** (__stack_t __stack)
- void **collect_warnings** (__warning_vector_t &__warnings)
- void **destruct** (__container_size_info *__obj_info, std::size_t __num, std::size_t __inum)
- void **free** ()
- [__container_size_info](#) * **insert** (__stack_t __stack, std::size_t __num)
- void **retire_object** (__container_size_info *__info)
- void **write** (FILE *__f)

Protected Attributes

- const char * **__id**

5.415.1 Detailed Description

Container size instrumentation trace producer.

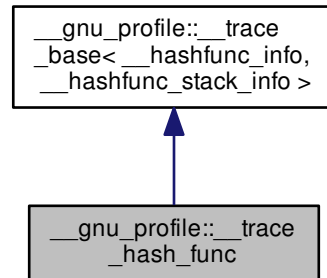
Definition at line 156 of file `profiler_container_size.h`.

The documentation for this class was generated from the following file:

- [profiler_container_size.h](#)

5.416 `__gnu_profile::__trace_hash_func` Class Reference

Inheritance diagram for `__gnu_profile::__trace_hash_func`:



Public Member Functions

- [`__hashfunc_info`](#) * **add_object** (`__stack_t` __stack)
- void **collect_warnings** (`__warning_vector_t` &__warnings)
- void **destruct** ([`__hashfunc_info`](#) * __obj_info, `std::size_t` __chain, `std::size_t` __accesses, `std::size_t` __hops)
- void **free** ()
- void **retire_object** ([`__hashfunc_info`](#) * __info)
- void **write** (`FILE` * __f)

Protected Attributes

- const char * **__id**

5.416.1 Detailed Description

Hash performance instrumentation producer.

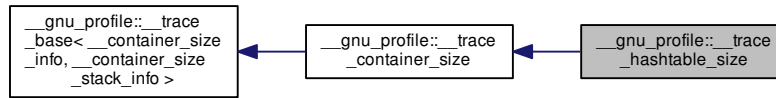
Definition at line 95 of file `profiler_hash_func.h`.

The documentation for this class was generated from the following file:

- [profiler_hash_func.h](#)

5.417 `__gnu_profile::__trace_hashtable_size` Class Reference

Inheritance diagram for `__gnu_profile::__trace_hashtable_size`:



Public Member Functions

- `__container_size_info * __add_object (__stack_t __stack)`
- `void __collect_warnings (__warning_vector_t &__warnings)`
- `void __destruct (__container_size_info *__obj_info, std::size_t __num, std::size_t __inum)`
- `void __free ()`
- `__container_size_info * __insert (__stack_t __stack, std::size_t __num)`
- `void __retire_object (__container_size_info *__info)`
- `void __write (FILE *__f)`

Protected Attributes

- `const char * __id`

5.417.1 Detailed Description

Hashtable size instrumentation trace producer.

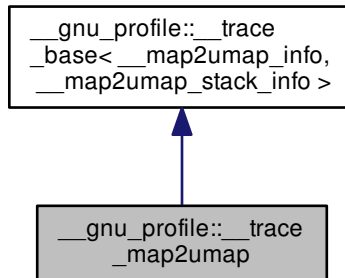
Definition at line 42 of file `profiler_hashtable_size.h`.

The documentation for this class was generated from the following file:

- [profiler_hashtable_size.h](#)

5.418 `__gnu_profile::__trace_map2umap` Class Reference

Inheritance diagram for `__gnu_profile::__trace_map2umap`:



Public Member Functions

- `__map2umap_info * __add_object (__stack_t __stack)`
- `void __collect_warnings (__warning_vector_t &__warnings)`
- `void __destruct (__map2umap_info *__obj_info)`
- `void __free ()`
- `void __retire_object (__map2umap_info *__info)`
- `void __write (FILE *__f)`

Protected Attributes

- `const char * __id`

5.418.1 Detailed Description

Map-to-unordered_map instrumentation producer.

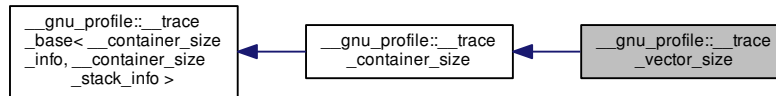
Definition at line 173 of file `profiler_map_to_unordered_map.h`.

The documentation for this class was generated from the following file:

- [profiler_map_to_unordered_map.h](#)

5.419 `__gnu_profile::__trace_vector_size` Class Reference

Inheritance diagram for `__gnu_profile::__trace_vector_size`:



Public Member Functions

- `__container_size_info * __add_object (__stack_t __stack)`
- `void __collect_warnings (__warning_vector_t &__warnings)`
- `void __destruct (__container_size_info *__obj_info, std::size_t __num, std::size_t __inum)`
- `void __free ()`
- `__container_size_info * __insert (__stack_t __stack, std::size_t __num)`
- `void __retire_object (__container_size_info *__info)`
- `void __write (FILE *__f)`

Protected Attributes

- `const char * __id`

5.419.1 Detailed Description

Hashtable size instrumentation trace producer.

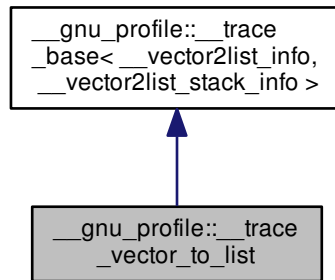
Definition at line 42 of file `profiler_vector_size.h`.

The documentation for this class was generated from the following file:

- [profiler_vector_size.h](#)

5.420 `__gnu_profile::__trace_vector_to_list` Class Reference

Inheritance diagram for `__gnu_profile::__trace_vector_to_list`:



Public Member Functions

- [__vector2list_info](#) * **__add_object** (`__stack_t` __stack)
- void **__collect_warnings** (`__warning_vector_t` &__warnings)
- void **__destruct** ([__vector2list_info](#) * __obj_info)
- void **__free** ()
- float **__list_cost** (`std::size_t` __shift, `std::size_t` __iterate, `std::size_t` __resize)
- void **__retire_object** ([__vector2list_info](#) * __info)
- float **__vector_cost** (`std::size_t` __shift, `std::size_t` __iterate, `std::size_t` __resize)
- void **__write** (FILE * __f)

Protected Attributes

- const char * **__id**

5.420.1 Detailed Description

Vector-to-list instrumentation producer.

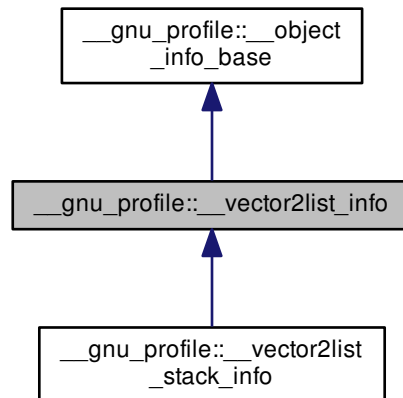
Definition at line 131 of file `profiler_vector_to_list.h`.

The documentation for this class was generated from the following file:

- [profiler_vector_to_list.h](#)

5.421 `__gnu_profile::__vector2list_info` Class Reference

Inheritance diagram for `__gnu_profile::__vector2list_info`:



Public Member Functions

- `__vector2list_info` (`__stack_t __stack`)
- `std::string __advice` () const
- `bool __is_valid` () const
- `std::size_t __iterate` ()
- `float __list_cost` ()
- `float __magnitude` () const
- `void __merge` (const `__vector2list_info` &`__o`)
- `void __merge` (const `__object_info_base` &`__o`)
- `void __opr_insert` (`std::size_t __pos`, `std::size_t __num`)
- `void __opr_iterate` (`int __num`)
- `std::size_t __resize` ()
- `void __resize` (`std::size_t __from`, `std::size_t`)
- `void __set_invalid` ()
- `void __set_list_cost` (`float __lc`)
- `void __set_vector_cost` (`float __vc`)
- `std::size_t __shift_count` ()
- `__stack_t __stack` () const
- `void __write` (`FILE *__f`) const

Protected Attributes

- `__stack_t __M_stack`
- `bool __M_valid`

5.421.1 Detailed Description

A vector-to-list instrumentation line in the object table.

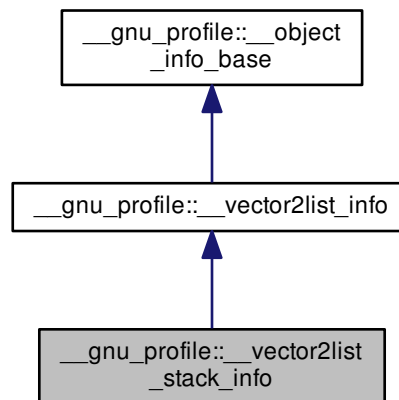
Definition at line 40 of file profiler_vector_to_list.h.

The documentation for this class was generated from the following file:

- [profiler_vector_to_list.h](#)

5.422 __gnu_profile::__vector2list_stack_info Class Reference

Inheritance diagram for __gnu_profile::__vector2list_stack_info:



Public Member Functions

- [__vector2list_stack_info](#) (const [__vector2list_info](#) &__o)
- [std::string __advice](#) () const
- [bool __is_valid](#) () const
- [std::size_t __iterate](#) ()
- [float __list_cost](#) ()
- [float __magnitude](#) () const
- [void __merge](#) (const [__vector2list_info](#) &__o)
- [void __merge](#) (const [__object_info_base](#) &__o)
- [void __opr_insert](#) (std::size_t __pos, std::size_t __num)
- [void __opr_iterate](#) (int __num)
- [std::size_t __resize](#) ()
- [void __resize](#) (std::size_t __from, std::size_t)
- [void __set_invalid](#) ()
- [void __set_list_cost](#) (float __lc)
- [void __set_vector_cost](#) (float __vc)
- [std::size_t __shift_count](#) ()
- [__stack_t __stack](#) () const
- [void __write](#) (FILE * __f) const

Protected Attributes

- `__stack_t __M_stack`
- `bool __M_valid`

5.422.1 Detailed Description

A vector-to-list instrumentation line in the stack table.

Definition at line 121 of file `profiler_vector_to_list.h`.

The documentation for this class was generated from the following file:

- [profiler_vector_to_list.h](#)

5.423 `__gnu_profile::__warning_data` Struct Reference

Public Member Functions

- `__warning_data` (float `__m`, `__stack_t` `__c`, const char *`__id`, const [std::string](#) &`__msg`)
- `bool operator<` (const [__warning_data](#) &`__other`) const

Public Attributes

- `__stack_t __context`
- `float __magnitude`
- `const char * __warning_id`
- [std::string](#) `__warning_message`

5.423.1 Detailed Description

Representation of a warning.

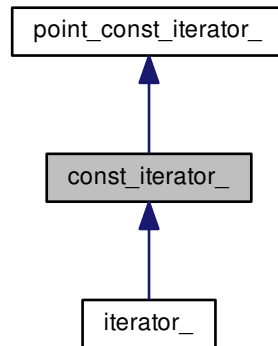
Definition at line 73 of file `profiler_trace.h`.

The documentation for this struct was generated from the following file:

- [profiler_trace.h](#)

5.424 `const_iterator_` Class Reference

Inheritance diagram for `const_iterator_`:



Public Types

- `typedef const_pointer_ const_pointer`
- `typedef const_reference_ const_reference`
- `typedef _Alloc::difference_type difference_type`
- `typedef std::forward_iterator_tag iterator_category`
- `typedef pointer_ pointer`
- `typedef reference_ reference`
- `typedef value_type_ value_type`

Public Member Functions

- `const_iterator_ ()`
- `bool operator!= (const point_iterator_ &other) const`
- `bool operator!= (const point_const_iterator_ &other) const`
- `const_reference operator* () const`
- `const_iterator_ & operator++ ()`
- `const_iterator_ operator++ (int)`
- `const_pointer operator-> () const`
- `bool operator== (const point_iterator_ &other) const`
- `bool operator== (const point_const_iterator_ &other) const`

Protected Types

- `typedef point_const_iterator_ base_type`

Protected Member Functions

- [const_iterator_](#) (const_pointer_ p_value, PB_DS_GEN_POS pos, const PB_DS_CLASS_C_DEC *p_tbl)

Protected Attributes

- const PB_DS_CLASS_C_DEC * [m_p_tbl](#)
- [const_pointer](#) **m_p_value**
- PB_DS_GEN_POS **m_pos**

Friends

- class **PB_DS_CLASS_C_DEC**

5.424.1 Detailed Description

Const range-type iterator.

Definition at line 43 of file unordered_iterator/const_iterator.hpp.

5.424.2 Member Typedef Documentation

5.424.2.1 `typedef const_pointer_ const_iterator_::const_pointer`

Iterator's const pointer type.

Definition at line 60 of file unordered_iterator/const_iterator.hpp.

5.424.2.2 `typedef const_reference_ const_iterator_::const_reference`

Iterator's const reference type.

Definition at line 66 of file unordered_iterator/const_iterator.hpp.

5.424.2.3 `typedef _Alloc::difference_type const_iterator_::difference_type`

Difference type.

Definition at line 51 of file unordered_iterator/const_iterator.hpp.

5.424.2.4 `typedef std::forward_iterator_tag const_iterator_::iterator_category`

Category.

Definition at line 48 of file unordered_iterator/const_iterator.hpp.

5.424.2.5 typedef pointer_const_iterator_::pointer

Iterator's pointer type.

Definition at line 57 of file unordered_iterator/const_iterator.hpp.

5.424.2.6 typedef reference_const_iterator_::reference

Iterator's reference type.

Definition at line 63 of file unordered_iterator/const_iterator.hpp.

5.424.2.7 typedef value_type_const_iterator_::value_type

Iterator's value type.

Definition at line 54 of file unordered_iterator/const_iterator.hpp.

5.424.3 Constructor & Destructor Documentation

5.424.3.1 const_iterator_::const_iterator_ () [inline]

Default constructor.

Definition at line 69 of file unordered_iterator/const_iterator.hpp.

5.424.3.2 const_iterator_::const_iterator_ (const_pointer_ p_value, PB_DS_GEN_POS pos, const PB_DS_CLASS_C_DEC * p_tbl) [inline], [protected]

Constructor used by the table to initiate the generalized pointer and position (e.g., this is called from within a find() of a table.

Definition at line 97 of file unordered_iterator/const_iterator.hpp.

5.424.4 Member Function Documentation

5.424.4.1 bool point_const_iterator_::operator!= (const point_iterator_ & other) const [inline], [inherited]

Compares content (negatively) to a different iterator object.

Definition at line 118 of file unordered_iterator/point_const_iterator.hpp.

5.424.4.2 bool point_const_iterator_::operator!= (const point_const_iterator_ & other) const [inline], [inherited]

Compares content (negatively) to a different iterator object.

Definition at line 123 of file unordered_iterator/point_const_iterator.hpp.

5.424.4.3 `const_reference point_const_iterator::operator*() const` `[inline],[inherited]`

Access.

Definition at line 100 of file `unordered_iterator/point_const_iterator.hpp`.

5.424.4.4 `const_iterator_ & const_iterator::operator++()` `[inline]`

Increments.

Definition at line 74 of file `unordered_iterator/const_iterator.hpp`.

References `m_p_tbl`.

5.424.4.5 `const_iterator_ const_iterator::operator++(int)` `[inline]`

Increments.

Definition at line 82 of file `unordered_iterator/const_iterator.hpp`.

References `m_p_tbl`.

5.424.4.6 `const_pointer point_const_iterator::operator->() const` `[inline],[inherited]`

Access.

Definition at line 92 of file `unordered_iterator/point_const_iterator.hpp`.

5.424.4.7 `bool point_const_iterator::operator==(const point_iterator_ & other) const` `[inline],[inherited]`

Compares content to a different iterator object.

Definition at line 108 of file `unordered_iterator/point_const_iterator.hpp`.

5.424.4.8 `bool point_const_iterator::operator==(const point_const_iterator_ & other) const` `[inline],[inherited]`

Compares content to a different iterator object.

Definition at line 113 of file `unordered_iterator/point_const_iterator.hpp`.

5.424.5 Member Data Documentation

5.424.5.1 `const PB_DS_CLASS_C_DEC* const_iterator::m_p_tbl` `[protected]`

Pointer to the table object which created the iterator (used for incrementing its position).

Definition at line 106 of file `unordered_iterator/const_iterator.hpp`.

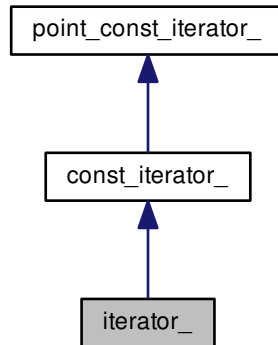
Referenced by `operator++()`, and `iterator::operator++()`.

The documentation for this class was generated from the following file:

- [unordered_iterator/const_iterator.hpp](#)

5.425 iterator_ Class Reference

Inheritance diagram for iterator_:



Public Types

- typedef const_pointer_ [const_pointer](#)
- typedef const_reference_ [const_reference](#)
- typedef _Alloc::difference_type [difference_type](#)
- typedef [std::forward_iterator_tag](#) [iterator_category](#)
- typedef pointer_ [pointer](#)
- typedef reference_ [reference](#)
- typedef value_type_ [value_type](#)

Public Member Functions

- [iterator_\(\)](#)
- [operator const point_iterator_\(\)](#) const
- [operator point_iterator_\(\)](#)
- bool [operator!=](#) (const [point_iterator_](#) &other) const
- bool [operator!=](#) (const [point_const_iterator_](#) &other) const
- [reference operator*](#) () const
- [iterator_ & operator++](#) ()
- [iterator_ operator++](#) (int)
- [pointer operator->](#) () const
- bool [operator==](#) (const [point_iterator_](#) &other) const
- bool [operator==](#) (const [point_const_iterator_](#) &other) const

Protected Types

- typedef [const_iterator_](#) **base_type**

Protected Member Functions

- `iterator_` (`pointer` p_value, PB_DS_GEN_POS pos, PB_DS_CLASS_C_DEC *p_tbl)

Protected Attributes

- `const` PB_DS_CLASS_C_DEC * `m_p_tbl`
- `const_pointer` `m_p_value`
- PB_DS_GEN_POS `m_pos`

Friends

- class **PB_DS_CLASS_C_DEC**

5.425.1 Detailed Description

Range-type iterator.

Definition at line 43 of file iterator.hpp.

5.425.2 Member Typedef Documentation

5.425.2.1 `typedef const_pointer_iterator_::const_pointer`

Iterator's const pointer type.

Definition at line 60 of file iterator.hpp.

5.425.2.2 `typedef const_reference_iterator_::const_reference`

Iterator's const reference type.

Definition at line 66 of file iterator.hpp.

5.425.2.3 `typedef _Alloc::difference_type iterator_::difference_type`

Difference type.

Definition at line 51 of file iterator.hpp.

5.425.2.4 `typedef std::forward_iterator_tag iterator_::iterator_category`

Category.

Definition at line 48 of file iterator.hpp.

5.425.2.5 `typedef pointer_iterator_::pointer`

Iterator's pointer type.

Definition at line 57 of file iterator.hpp.

5.425.2.6 `typedef reference_iterator_::reference`

Iterator's reference type.

Definition at line 63 of file iterator.hpp.

5.425.2.7 `typedef value_type_iterator_::value_type`

Iterator's value type.

Definition at line 54 of file iterator.hpp.

5.425.3 Constructor & Destructor Documentation

5.425.3.1 `iterator_::iterator_() [inline]`

Default constructor.

Definition at line 70 of file iterator.hpp.

5.425.3.2 `iterator_::iterator_(pointer p_value, PB_DS_GEN_POS pos, PB_DS_CLASS_C_DEC * p_tbl) [inline], [protected]`

Constructor used by the table to initiate the generalized pointer and position (e.g., this is called from within a find() of a table.

Definition at line 125 of file iterator.hpp.

5.425.4 Member Function Documentation

5.425.4.1 `iterator_::operator const point_iterator_() const [inline]`

Conversion to a point-type iterator.

Definition at line 80 of file iterator.hpp.

5.425.4.2 `iterator_::operator point_iterator_() [inline]`

Conversion to a point-type iterator.

Definition at line 75 of file iterator.hpp.

5.425.4.3 `bool point_const_iterator::operator!=(const point_iterator_ & other) const` `[inline]`, `[inherited]`

Compares content (negatively) to a different iterator object.

Definition at line 118 of file `unordered_iterator/point_const_iterator.hpp`.

5.425.4.4 `bool point_const_iterator::operator!=(const point_const_iterator_ & other) const` `[inline]`, `[inherited]`

Compares content (negatively) to a different iterator object.

Definition at line 123 of file `unordered_iterator/point_const_iterator.hpp`.

5.425.4.5 `reference iterator::operator*() const` `[inline]`

Access.

Definition at line 93 of file `iterator.hpp`.

5.425.4.6 `iterator_ & iterator::operator++()` `[inline]`

Increments.

Definition at line 101 of file `iterator.hpp`.

References `const_iterator::m_p_tbl`.

5.425.4.7 `iterator_iterator::operator++(int)` `[inline]`

Increments.

Definition at line 109 of file `iterator.hpp`.

References `const_iterator::m_p_tbl`.

5.425.4.8 `pointer iterator::operator->() const` `[inline]`

Access.

Definition at line 85 of file `iterator.hpp`.

5.425.4.9 `bool point_const_iterator::operator==(const point_iterator_ & other) const` `[inline]`, `[inherited]`

Compares content to a different iterator object.

Definition at line 108 of file `unordered_iterator/point_const_iterator.hpp`.

5.425.4.10 `bool point_const_iterator_::operator== (const point_const_iterator_ & other) const` `[inline]`,
`[inherited]`

Compares content to a different iterator object.

Definition at line 113 of file `unordered_iterator/point_const_iterator.hpp`.

5.425.5 Member Data Documentation

5.425.5.1 `const PB_DS_CLASS_C_DEC* const_iterator_::m_p_tbl` `[protected]`, `[inherited]`

Pointer to the table object which created the iterator (used for incrementing its position).

Definition at line 106 of file `unordered_iterator/const_iterator.hpp`.

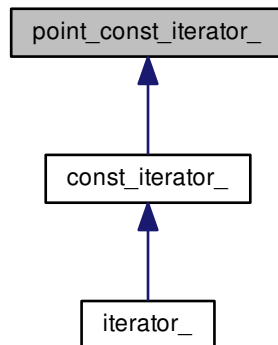
Referenced by `const_iterator_::operator++()`, and `operator++()`.

The documentation for this class was generated from the following file:

- [iterator.hpp](#)

5.426 point_const_iterator_ Class Reference

Inheritance diagram for `point_const_iterator_`:



Public Types

- typedef const_pointer_ [const_pointer](#)
- typedef const_reference_ [const_reference](#)
- typedef trivial_iterator_difference_type [difference_type](#)
- typedef trivial_iterator_tag [iterator_category](#)
- typedef pointer_ [pointer](#)
- typedef reference_ [reference](#)
- typedef value_type_ [value_type](#)

Public Member Functions

- **point_const_iterator_** ([const_pointer](#) p_value)
- [point_const_iterator_](#) ()
- [point_const_iterator_](#) (const [point_const_iterator_](#) &other)
- [point_const_iterator_](#) (const [point_iterator_](#) &other)
- bool [operator!=](#) (const [point_iterator_](#) &other) const
- bool [operator!=](#) (const [point_const_iterator_](#) &other) const
- [const_reference](#) [operator*](#) () const
- [const_pointer](#) [operator->](#) () const
- bool [operator==](#) (const [point_iterator_](#) &other) const
- bool [operator==](#) (const [point_const_iterator_](#) &other) const

Protected Attributes

- [const_pointer](#) **m_p_value**

Friends

- class **PB_DS_CLASS_C_DEC**
- class **point_iterator_**

5.426.1 Detailed Description

Const point-type iterator.

Definition at line 45 of file unordered_iterator/point_const_iterator.hpp.

5.426.2 Member Typedef Documentation

5.426.2.1 typedef const_pointer_ point_const_iterator_::const_pointer

Iterator's const pointer type.

Definition at line 61 of file unordered_iterator/point_const_iterator.hpp.

5.426.2.2 `typedef const_reference point_const_iterator::const_reference`

Iterator's const reference type.

Definition at line 67 of file `unordered_iterator/point_const_iterator.hpp`.

5.426.2.3 `typedef trivial_iterator_difference_type point_const_iterator::difference_type`

Difference type.

Definition at line 52 of file `unordered_iterator/point_const_iterator.hpp`.

5.426.2.4 `typedef trivial_iterator_tag point_const_iterator::iterator_category`

Category.

Definition at line 49 of file `unordered_iterator/point_const_iterator.hpp`.

5.426.2.5 `typedef pointer point_const_iterator::pointer`

Iterator's pointer type.

Definition at line 58 of file `unordered_iterator/point_const_iterator.hpp`.

5.426.2.6 `typedef reference point_const_iterator::reference`

Iterator's reference type.

Definition at line 64 of file `unordered_iterator/point_const_iterator.hpp`.

5.426.2.7 `typedef value_type point_const_iterator::value_type`

Iterator's value type.

Definition at line 55 of file `unordered_iterator/point_const_iterator.hpp`.

5.426.3 Constructor & Destructor Documentation

5.426.3.1 `point_const_iterator::point_const_iterator()` `[inline]`

Default constructor.

Definition at line 75 of file `unordered_iterator/point_const_iterator.hpp`.

5.426.3.2 `point_const_iterator::point_const_iterator(const point_const_iterator_ & other)` `[inline]`

Copy constructor.

Definition at line 80 of file `unordered_iterator/point_const_iterator.hpp`.

5.426.3.3 `point_const_iterator::point_const_iterator_ (const point_iterator_ & other)` `[inline]`

Copy constructor.

Definition at line 86 of file `unordered_iterator/point_const_iterator.hpp`.

5.426.4 Member Function Documentation

5.426.4.1 `bool point_const_iterator::operator!= (const point_iterator_ & other) const` `[inline]`

Compares content (negatively) to a different iterator object.

Definition at line 118 of file `unordered_iterator/point_const_iterator.hpp`.

5.426.4.2 `bool point_const_iterator::operator!= (const point_const_iterator_ & other) const` `[inline]`

Compares content (negatively) to a different iterator object.

Definition at line 123 of file `unordered_iterator/point_const_iterator.hpp`.

5.426.4.3 `const_reference point_const_iterator::operator* () const` `[inline]`

Access.

Definition at line 100 of file `unordered_iterator/point_const_iterator.hpp`.

5.426.4.4 `const_pointer point_const_iterator::operator-> () const` `[inline]`

Access.

Definition at line 92 of file `unordered_iterator/point_const_iterator.hpp`.

5.426.4.5 `bool point_const_iterator::operator== (const point_iterator_ & other) const` `[inline]`

Compares content to a different iterator object.

Definition at line 108 of file `unordered_iterator/point_const_iterator.hpp`.

5.426.4.6 `bool point_const_iterator::operator== (const point_const_iterator_ & other) const` `[inline]`

Compares content to a different iterator object.

Definition at line 113 of file `unordered_iterator/point_const_iterator.hpp`.

The documentation for this class was generated from the following file:

- [unordered_iterator/point_const_iterator.hpp](#)

5.427 point_iterator_ Class Reference

Public Types

- typedef const_pointer_ [const_pointer](#)
- typedef const_reference_ [const_reference](#)
- typedef trivial_iterator_difference_type [difference_type](#)
- typedef trivial_iterator_tag [iterator_category](#)
- typedef pointer_ [pointer](#)
- typedef reference_ [reference](#)
- typedef value_type_ [value_type](#)

Public Member Functions

- [point_iterator_](#) ()
- [point_iterator_](#) (const [point_iterator_](#) &other)
- **point_iterator_** ([pointer](#) p_value)
- bool [operator!=](#) (const [point_iterator_](#) &other) const
- bool [operator!=](#) (const [point_const_iterator_](#) &other) const
- [reference](#) [operator*](#) () const
- [pointer](#) [operator->](#) () const
- bool [operator==](#) (const [point_iterator_](#) &other) const
- bool [operator==](#) (const [point_const_iterator_](#) &other) const

Protected Attributes

- [pointer](#) **m_p_value**

Friends

- class **PB_DS_CLASS_C_DEC**
- class **point_const_iterator_**

5.427.1 Detailed Description

Find type iterator.

Definition at line 43 of file point_iterator.hpp.

5.427.2 Member Typedef Documentation

5.427.2.1 typedef const_pointer_ point_iterator_::const_pointer

Iterator's const pointer type.

Definition at line 59 of file point_iterator.hpp.

5.427.2.2 `typedef const_reference_point_iterator::const_reference`

Iterator's const reference type.

Definition at line 65 of file `point_iterator.hpp`.

5.427.2.3 `typedef trivial_iterator_difference_type_point_iterator::difference_type`

Difference type.

Definition at line 50 of file `point_iterator.hpp`.

5.427.2.4 `typedef trivial_iterator_tag_point_iterator::iterator_category`

Category.

Definition at line 47 of file `point_iterator.hpp`.

5.427.2.5 `typedef pointer_point_iterator::pointer`

Iterator's pointer type.

Definition at line 56 of file `point_iterator.hpp`.

5.427.2.6 `typedef reference_point_iterator::reference`

Iterator's reference type.

Definition at line 62 of file `point_iterator.hpp`.

5.427.2.7 `typedef value_type_point_iterator::value_type`

Iterator's value type.

Definition at line 53 of file `point_iterator.hpp`.

5.427.3 Constructor & Destructor Documentation

5.427.3.1 `point_iterator::point_iterator() [inline]`

Default constructor.

Definition at line 69 of file `point_iterator.hpp`.

Referenced by operator!=().

5.427.3.2 point_iterator::point_iterator_ (const point_iterator_ & other) [inline]

Copy constructor.

Definition at line 75 of file point_iterator.hpp.

5.427.4 Member Function Documentation

5.427.4.1 bool point_iterator::operator!= (const point_iterator_ & other) const [inline]

Compares content to a different iterator object.

Definition at line 107 of file point_iterator.hpp.

5.427.4.2 bool point_iterator::operator!= (const point_const_iterator_ & other) const [inline]

Compares content (negatively) to a different iterator object.

Definition at line 112 of file point_iterator.hpp.

References point_iterator_().

5.427.4.3 reference point_iterator::operator* () const [inline]

Access.

Definition at line 89 of file point_iterator.hpp.

5.427.4.4 pointer point_iterator::operator-> () const [inline]

Access.

Definition at line 81 of file point_iterator.hpp.

5.427.4.5 bool point_iterator::operator== (const point_iterator_ & other) const [inline]

Compares content to a different iterator object.

Definition at line 97 of file point_iterator.hpp.

5.427.4.6 bool point_iterator::operator== (const point_const_iterator_ & other) const [inline]

Compares content to a different iterator object.

Definition at line 102 of file point_iterator.hpp.

The documentation for this class was generated from the following file:

- [point_iterator.hpp](#)

5.428 std::__allocated_ptr<_Alloc> Struct Template Reference

Public Types

- using **pointer** = typename [allocator_traits](#)<_Alloc>::pointer
- using **value_type** = typename [allocator_traits](#)<_Alloc>::value_type

Public Member Functions

- [__allocated_ptr](#) (_Alloc &__a, pointer __ptr) noexcept
- template<typename _Ptr, typename _Req = _Require<is_same<_Ptr, value_type*>>>
 [__allocated_ptr](#) (_Alloc &__a, _Ptr __ptr)
- [__allocated_ptr](#) ([__allocated_ptr](#) &&__gd) noexcept
- [~__allocated_ptr](#) ()
- value_type * [get](#) ()
- [__allocated_ptr](#) & [operator=](#) (std::nullptr_t) noexcept

5.428.1 Detailed Description

```
template<typename _Alloc>
struct std::__allocated_ptr<_Alloc>
```

Non-standard RAll type for managing pointers obtained from allocators.

Definition at line 46 of file `allocated_ptr.h`.

5.428.2 Constructor & Destructor Documentation

5.428.2.1 `template<typename _Alloc> std::__allocated_ptr<_Alloc>::__allocated_ptr (_Alloc & __a, pointer __ptr)`
`[inline], [noexcept]`

Take ownership of __ptr.

Definition at line 52 of file `allocated_ptr.h`.

5.428.2.2 `template<typename _Alloc> template<typename _Ptr, typename _Req = _Require<is_same<_Ptr, value_type*>>>
std::__allocated_ptr<_Alloc>::__allocated_ptr (_Alloc & __a, _Ptr __ptr)` `[inline]`

Convert __ptr to allocator's pointer type and take ownership of it.

Definition at line 59 of file `allocated_ptr.h`.

5.428.2.3 `template<typename _Alloc> std::__allocated_ptr<_Alloc>::__allocated_ptr (__allocated_ptr<_Alloc>
&& __gd)` `[inline], [noexcept]`

Transfer ownership of the owned pointer.

Definition at line 65 of file `allocated_ptr.h`.

5.428.2.4 `template<typename _Alloc> std::__allocated_ptr<_Alloc>::~~__allocated_ptr() [inline]`

Deallocate the owned pointer.

Definition at line 70 of file `allocated_ptr.h`.

References `std::allocator_traits<_Alloc>::deallocate()`.

5.428.3 Member Function Documentation

5.428.3.1 `template<typename _Alloc> value_type* std::__allocated_ptr<_Alloc>::get(void) [inline]`

Get the address that the owned pointer refers to.

Definition at line 85 of file `allocated_ptr.h`.

5.428.3.2 `template<typename _Alloc> __allocated_ptr& std::__allocated_ptr<_Alloc>::operator=(std::nullptr_t) [inline], [noexcept]`

Release ownership of the owned pointer.

Definition at line 78 of file `allocated_ptr.h`.

The documentation for this struct was generated from the following file:

- [allocated_ptr.h](#)

5.429 std::__atomic_base<_ITp> Struct Template Reference

Public Member Functions

- `__atomic_base` (const `__atomic_base` &)=delete
- `constexpr __atomic_base` (`__int_type` __i) noexcept
- `__attribute__((always_inline)) void store` (`__int_type` __i
- `bool is_lock_free` () const noexcept
- `bool is_lock_free` () const volatile noexcept
- `operator __int_type` () const noexcept
- `operator __int_type` () const volatile noexcept
- `__int_type operator&=` (`__int_type` __i) noexcept
- `__int_type operator&=` (`__int_type` __i) volatile noexcept
- `__int_type operator++` (int) noexcept
- `__int_type operator++` (int) volatile noexcept
- `__int_type operator++` () noexcept
- `__int_type operator++` () volatile noexcept
- `__int_type operator+=` (`__int_type` __i) noexcept
- `__int_type operator+=` (`__int_type` __i) volatile noexcept
- `__int_type operator--` (int) noexcept
- `__int_type operator--` (int) volatile noexcept

- `__int_type operator-- () noexcept`
- `__int_type operator-- () volatile noexcept`
- `__int_type operator= (__int_type __i) noexcept`
- `__int_type operator= (__int_type __i) volatile noexcept`
- `__atomic_base & operator= (const __atomic_base &)=delete`
- `__atomic_base & operator= (const __atomic_base &) volatile=delete`
- `__int_type operator= (__int_type __i) noexcept`
- `__int_type operator= (__int_type __i) volatile noexcept`
- `__int_type operator^= (__int_type __i) noexcept`
- `__int_type operator^= (__int_type __i) volatile noexcept`
- `__int_type operator|= (__int_type __i) noexcept`
- `__int_type operator|= (__int_type __i) volatile noexcept`

5.429.1 Detailed Description

```
template<typename _ITp>
struct std::__atomic_base< _ITp >
```

Base class for atomic integrals.

Definition at line 120 of file `atomic_base.h`.

The documentation for this struct was generated from the following file:

- [atomic_base.h](#)

5.430 `std::__atomic_base< _PTp * >` Struct Template Reference

Public Member Functions

- `__atomic_base (const __atomic_base &)=delete`
- `constexpr __atomic_base (__pointer_type __p) noexcept`
- `__attribute__((always_inline)) void store(__pointer_type __p`
- `bool is_lock_free () const noexcept`
- `bool is_lock_free () const volatile noexcept`
- `operator __pointer_type () const noexcept`
- `operator __pointer_type () const volatile noexcept`
- `__pointer_type operator++ (int) noexcept`
- `__pointer_type operator++ (int) volatile noexcept`
- `__pointer_type operator++ () noexcept`
- `__pointer_type operator++ () volatile noexcept`
- `__pointer_type operator+= (ptrdiff_t __d) noexcept`
- `__pointer_type operator+= (ptrdiff_t __d) volatile noexcept`
- `__pointer_type operator-- (int) noexcept`
- `__pointer_type operator-- (int) volatile noexcept`
- `__pointer_type operator-- () noexcept`
- `__pointer_type operator-- () volatile noexcept`
- `__pointer_type operator-= (ptrdiff_t __d) noexcept`
- `__pointer_type operator-= (ptrdiff_t __d) volatile noexcept`
- `__atomic_base & operator= (const __atomic_base &)=delete`
- `__atomic_base & operator= (const __atomic_base &) volatile=delete`
- `__pointer_type operator= (__pointer_type __p) noexcept`
- `__pointer_type operator= (__pointer_type __p) volatile noexcept`

5.430.1 Detailed Description

```
template<typename _PTp>
struct std::__atomic_base< _PTp * >
```

Partial specialization for pointer types.

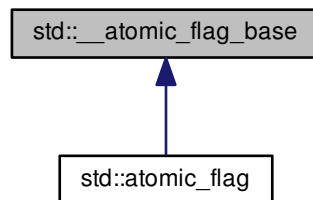
Definition at line 565 of file atomic_base.h.

The documentation for this struct was generated from the following file:

- [atomic_base.h](#)

5.431 std::__atomic_flag_base Struct Reference

Inheritance diagram for std::__atomic_flag_base:



Public Attributes

- __atomic_flag_data_type **M** i

5.431.1 Detailed Description

Base type for atomic_flag.

Base type is POD with data, allowing atomic_flag to derive from it and meet the standard layout type requirement. In addition to compatibility with a C interface, this allows different implementations of atomic_flag to use the same atomic operation functions, via a standard conversion to the __atomic_flag_base argument.

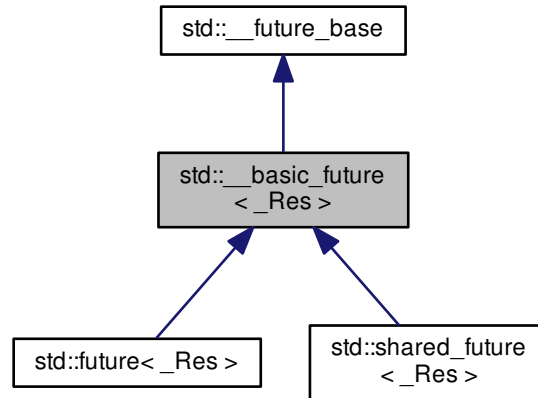
Definition at line 150 of file atomic_base.h.

The documentation for this struct was generated from the following file:

- [atomic_base.h](#)

5.432 `std::__basic_future<_Res>` Class Template Reference

Inheritance diagram for `std::__basic_future<_Res>`:



Public Types

- `template<typename _Res>`
`using _Ptr = unique_ptr<_Res, _Result_base::_Deleter>`
- `using _State_base = _State_baseV2`

Public Member Functions

- `__basic_future (const __basic_future &)=delete`
- `__basic_future & operator= (const __basic_future &)=delete`
- `bool valid () const noexcept`
- `void wait () const`
- `template<typename _Rep, typename _Period>`
`future_status wait_for (const chrono::duration<_Rep, _Period> &__rel) const`
- `template<typename _Clock, typename _Duration>`
`future_status wait_until (const chrono::time_point<_Clock, _Duration> &__abs) const`

Static Public Member Functions

- `template<typename _Res, typename _Allocator>`
`static _Ptr<_Result_alloc<_Res, _Allocator>> _S_allocate_result (const _Allocator &__a)`
- `template<typename _Res, typename _Tp>`
`static _Ptr<_Result<_Res>> _S_allocate_result (const std::allocator<_Tp> &__a)`
- `template<typename _BoundFn>`
`static std::shared_ptr<_State_base> _S_make_async_state (_BoundFn &&__fn)`
- `template<typename _BoundFn>`
`static std::shared_ptr<_State_base> _S_make_deferred_state (_BoundFn &&__fn)`
- `template<typename _Res_ptr, typename _BoundFn>`
`static _Task_setter<_Res_ptr, _BoundFn> _S_task_setter (_Res_ptr &__ptr, _BoundFn &__call)`

Protected Types

- typedef [__future_base::__Result](#)<_Res> & [__result_type](#)
- typedef [shared_ptr](#)<_State_base> [__state_type](#)

Protected Member Functions

- [__basic_future](#) (const [__state_type](#) &_state)
- [__basic_future](#) (const [shared_future](#)<_Res> &) noexcept
- [__basic_future](#) ([shared_future](#)<_Res> &&) noexcept
- [__basic_future](#) ([future](#)<_Res> &&) noexcept
- [__result_type](#) [_M_get_result](#) () const
- void [_M_swap](#) ([__basic_future](#) &_that) noexcept

5.432.1 Detailed Description

```
template<typename _Res>
class std::__basic_future<_Res>
```

Common implementation for future and shared_future.

Definition at line 641 of file future.

5.432.2 Member Typedef Documentation

5.432.2.1 `template<typename _Res> using std::__future_base::__Ptr = unique_ptr<_Res, _Result_base::__Deleter>`
 [inherited]

A `unique_ptr` for result objects.

Definition at line 214 of file future.

5.432.3 Member Function Documentation

5.432.3.1 `template<typename _Res> __result_type std::__basic_future<_Res>::__M_get_result () const`
 [inline], [protected]

Wait for the state to be ready and rethrow any stored exception.

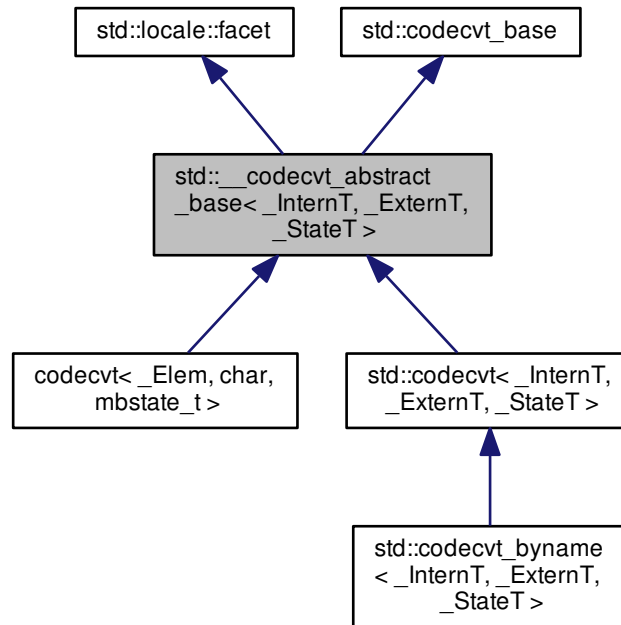
Definition at line 684 of file future.

The documentation for this class was generated from the following file:

- [future](#)

5.433 `std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >` Class Template Reference

Inheritance diagram for `std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >`:



Public Types

- typedef `_ExternT` **extern_type**
- typedef `_InternT` **intern_type**
- typedef `codecvt_base::result` **result**
- typedef `_StateT` **state_type**

Public Member Functions

- bool **always_noconv** () const throw ()
- int **encoding** () const throw ()
- result **in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- int **length** (state_type &__state, const extern_type *__from, const extern_type *__end, size_t __max) const
- int **max_length** () const throw ()
- result **out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- result **unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const

Protected Member Functions

- **__codecvt_abstract_base** (size_t __refs=0)
- virtual bool **do_always_noconv** () const =0 throw ()
- virtual int **do_encoding** () const =0 throw ()
- virtual result **do_in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *&__from_next, intern_type *__to, intern_type *__to_end, intern_type *&__to_next) const =0
- virtual int **do_length** (state_type &, const extern_type *__from, const extern_type *__end, size_t __max) const =0
- virtual int **do_max_length** () const =0 throw ()
- virtual result **do_out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *&__from_next, extern_type *__to, extern_type *__to_end, extern_type *&__to_next) const =0
- virtual result **do_unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *&__to↔__next) const =0

Static Protected Member Functions

- static __c_locale **_S_clone_c_locale** (__c_locale &__cloc) throw ()
- static void **_S_create_c_locale** (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale &__cloc)
- static __c_locale **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static __c_locale **_S_lc_ctype_c_locale** (__c_locale __cloc, const char *__s)

5.433.1 Detailed Description

```
template<typename _InternT, typename _ExternT, typename _StateT>
class std::__codecvt_abstract_base<_InternT, _ExternT, _StateT>
```

Common base for codecvt functions.

This template class provides implementations of the public functions that forward to the protected virtual functions.

This template also provides abstract stubs for the protected virtual functions.

Definition at line 68 of file codecvt.h.

5.433.2 Member Function Documentation

5.433.2.1 `template<typename _InternT, typename _ExternT, typename _StateT> virtual result std::__codecvt_abstract_base<_InternT, _ExternT, _StateT>::do_out (state_type & __state, const intern_type * __from, const intern_type * __from_end, const intern_type *& __from_next, extern_type * __to, extern_type * __to_end, extern_type *& __to_next) const [protected], [pure virtual]`

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This function is a hook for derived classes to change the value returned.

See also

out for more information.

Implemented in `std::codecvt< char32_t, char, mbstate_t >`, `std::codecvt< char16_t, char, mbstate_t >`, `std::codecvt< wchar_t, char, mbstate_t >`, `std::codecvt< char, char, mbstate_t >`, `std::codecvt< _InternT, _ExternT, _StateT >`, `std::codecvt< _Elem, char, mbstate_t >`, and `std::codecvt< _InternT, _ExternT, encoding_state >`.


```
5.433.2.2  template<typename _InternT, typename _ExternT, typename _StateT> result std::__codecvt_abstract_base<
    _InternT, _ExternT, _StateT >::in ( state_type & __state, const extern_type * __from, const extern_type * __from_end,
    const extern_type * & __from_next, intern_type * __to, intern_type * __to_end, intern_type * & __to_next ) const
    [inline]
```

Convert from external to internal character set.

Converts input string of `extern_type` to output string of `intern_type`. This is analogous to `mbsrtowcs`. It does this by calling `codecvt::do_in`.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in `[from,from_end)` are converted and written to `[to,to_end)`. `from_next` and `to_next` are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, `from_next` and `to_next` are not affected.

The `state` argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how `state` is used.

The result returned is a member of `codecvt_base::result`. If all the input is converted, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the input ends early or there is insufficient space in the output, returns `codecvt_base::partial`. Otherwise the conversion failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

Definition at line 196 of file `codecvt.h`.

```
5.433.2.3  template<typename _InternT, typename _ExternT, typename _StateT> result std::__codecvt_abstract_base<
    _InternT, _ExternT, _StateT >::out ( state_type & __state, const intern_type * __from, const intern_type * __from_end,
    const intern_type * & __from_next, extern_type * __to, extern_type * __to_end, extern_type * & __to_next ) const
    [inline]
```

Convert from internal to external character set.

Converts input string of `intern_type` to output string of `extern_type`. This is analogous to `wcsrtombs`. It does this by calling `codecvt::do_out`.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in `[from,from_end)` are converted and written to `[to,to_end)`. `from_next` and `to_next` are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, `from_next` and `to_next` are not affected.

The *state* argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how *state* is used.

The result returned is a member of `codecvt_base::result`. If all the input is converted, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the input ends early or there is insufficient space in the output, returns `codecvt_base::partial`. Otherwise the conversion failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

Definition at line 116 of file `codecvt.h`.

5.433.2.4 `template<typename _InternT, typename _ExternT, typename _StateT> result std::__codecvt_abstract_base<_InternT, _ExternT, _StateT>::unshift (state_type & __state, extern_type * __to, extern_type * __to_end, extern_type *& __to_next) const [inline]`

Reset conversion state.

Writes characters to output that would restore *state* to initial conditions. The idea is that if a partial conversion occurs, then the converting the characters written by this function would leave the state in initial conditions, rather than partial conversion state. It does this by calling `codecvt::do_unshift()`.

For example, if 4 external characters always converted to 1 internal character, and input to `in()` had 6 external characters with state saved, this function would write two characters to the output and set the state to initialized conditions.

The source and destination character sets are determined by the facet's locale, internal and external types.

The result returned is a member of `codecvt_base::result`. If the state could be reset and data written, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the output has insufficient space, returns `codecvt_base::partial`. Otherwise the reset failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

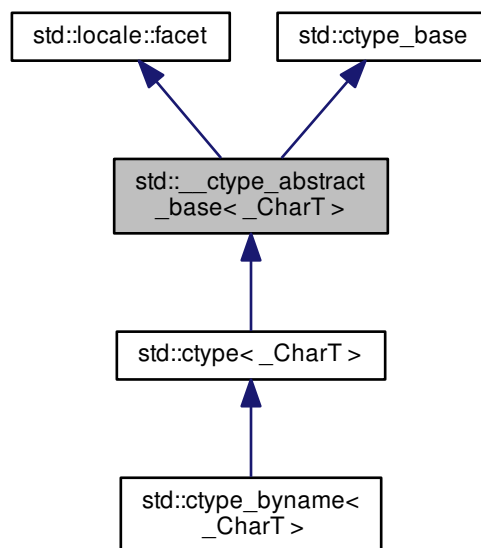
Definition at line 155 of file `codecvt.h`.

The documentation for this class was generated from the following file:

- [codecvt.h](#)

5.434 `std::__ctype_abstract_base<_CharT>` Class Template Reference

Inheritance diagram for `std::__ctype_abstract_base<_CharT>`:

**Public Types**

- `typedef const int * __to_type`
- `typedef _CharT char_type`
- `typedef unsigned short mask`

Public Member Functions

- bool **is** (mask __m, char_type __c) const
- const char_type * **is** (const char_type * __lo, const char_type * __hi, mask * __vec) const
- char **narrow** (char_type __c, char __default) const
- const char_type * **narrow** (const char_type * __lo, const char_type * __hi, char __default, char * __to) const
- const char_type * **scan_is** (mask __m, const char_type * __lo, const char_type * __hi) const
- const char_type * **scan_not** (mask __m, const char_type * __lo, const char_type * __hi) const
- char_type **tolower** (char_type __c) const
- const char_type * **tolower** (char_type * __lo, const char_type * __hi) const
- char_type **toupper** (char_type __c) const
- const char_type * **toupper** (char_type * __lo, const char_type * __hi) const
- char_type **widen** (char __c) const
- const char * **widen** (const char * __lo, const char * __hi, char_type * __to) const

Static Public Attributes

- static const mask **alnum**
- static const mask **alpha**
- static const mask **blank**
- static const mask **cntrl**
- static const mask **digit**
- static const mask **graph**
- static const mask **lower**
- static const mask **print**
- static const mask **punct**
- static const mask **space**
- static const mask **upper**
- static const mask **xdigit**

Protected Member Functions

- **__ctype_abstract_base** (size_t __refs=0)
- virtual bool **do_is** (mask __m, char_type __c) const =0
- virtual const char_type * **do_is** (const char_type * __lo, const char_type * __hi, mask * __vec) const =0
- virtual char **do_narrow** (char_type __c, char __default) const =0
- virtual const char_type * **do_narrow** (const char_type * __lo, const char_type * __hi, char __default, char * __to) const =0
- virtual const char_type * **do_scan_is** (mask __m, const char_type * __lo, const char_type * __hi) const =0
- virtual const char_type * **do_scan_not** (mask __m, const char_type * __lo, const char_type * __hi) const =0
- virtual char_type **do_tolower** (char_type __c) const =0
- virtual const char_type * **do_tolower** (char_type * __lo, const char_type * __hi) const =0
- virtual char_type **do_toupper** (char_type __c) const =0
- virtual const char_type * **do_toupper** (char_type * __lo, const char_type * __hi) const =0
- virtual char_type **do_widen** (char __c) const =0
- virtual const char * **do_widen** (const char * __lo, const char * __hi, char_type * __to) const =0

Static Protected Member Functions

- static `__c_locale _S_clone_c_locale (__c_locale &__cloc) throw ()`
- static void `_S_create_c_locale (__c_locale &__cloc, const char *__s, __c_locale __old=0)`
- static void `_S_destroy_c_locale (__c_locale &__cloc)`
- static `__c_locale _S_get_c_locale ()`
- static const char * `_S_get_c_name () throw ()`
- static `__c_locale _S_lc_ctype_c_locale (__c_locale __cloc, const char *__s)`

5.434.1 Detailed Description

```
template<typename _CharT>
class std::__ctype_abstract_base< _CharT >
```

Common base for ctype facet.

This template class provides implementations of the public functions that forward to the protected virtual functions.

This template also provides abstract stubs for the protected virtual functions.

Definition at line 150 of file locale_facets.h.

5.434.2 Member Typedef Documentation

5.434.2.1 `template<typename _CharT> typedef _CharT std::__ctype_abstract_base< _CharT >::char_type`

Typedef for the template parameter.

Definition at line 155 of file locale_facets.h.

5.434.3 Member Function Documentation

5.434.3.1 `template<typename _CharT> virtual bool std::__ctype_abstract_base< _CharT >::do_is (mask __m, char_type __c) const [protected], [pure virtual]`

Test char_type classification.

This function finds a mask M for *c* and compares it to mask *m*.

do_is() is a hook for a derived facet to change the behavior of classifying. do_is() must always return the same result for the same input.

Parameters

<code>__c</code>	The char_type to find the mask of.
<code>__m</code>	The mask to compare against.

Returns

(M & __m) != 0.

Implemented in [std::ctype<wchar_t>](#), and [std::ctype<_CharT>](#).

Referenced by [std::__ctype_abstract_base<wchar_t>::is\(\)](#), and [std::__ctype_abstract_base<wchar_t>::narrow\(\)](#).

5.434.3.2 `template<typename _CharT> virtual const char_type* std::__ctype_abstract_base<_CharT>::do_is (const char_type * __lo, const char_type * __hi, mask * __vec) const` [protected], [pure virtual]

Return a mask array.

This function finds the mask for each char_type in the range [lo,hi) and successively writes it to vec. vec must have as many elements as the input.

do_is() is a hook for a derived facet to change the behavior of classifying. do_is() must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__vec</code>	Pointer to an array of mask storage.

Returns

`__hi`.

Implemented in [std::ctype<wchar_t>](#), and [std::ctype<_CharT>](#).

5.434.3.3 `template<typename _CharT> virtual char std::__ctype_abstract_base<_CharT>::do_narrow (char_type __c, char __dfault) const` [protected], [pure virtual]

Narrow char_type to char.

This virtual function converts the argument to char using the simplest reasonable transformation. If the conversion fails, dfault is returned instead.

do_narrow() is a hook for a derived facet to change the behavior of narrowing. do_narrow() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See [codecvt](#) for that.

Parameters

<code>__c</code>	The char_type to convert.
<code>__dfault</code>	Char to return if conversion fails.

Returns

The converted char.

Implemented in [std::ctype< wchar_t >](#), and [std::ctype< _CharT >](#).

Referenced by `std::__ctype_abstract_base< wchar_t >::narrow()`, and `std::ctype< char >::narrow()`.

5.434.3.4 `template<typename _CharT> virtual const char_type* std::__ctype_abstract_base< _CharT >::do_narrow (const char_type * __lo, const char_type * __hi, char __dfault, char * __to) const [protected], [pure virtual]`

Narrow char_type array to char.

This virtual function converts each char_type in the range [`__lo`,`__hi`) to char using the simplest reasonable transformation and writes the results to the destination array. For any element in the input that cannot be converted, `__dfault` is used instead.

`do_narrow()` is a hook for a derived facet to change the behavior of narrowing. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__dfault</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Implemented in [std::ctype< wchar_t >](#), and [std::ctype< _CharT >](#).

5.434.3.5 `template<typename _CharT> virtual const char_type* std::__ctype_abstract_base< _CharT >::do_scan_is (mask __m, const char_type * __lo, const char_type * __hi) const [protected], [pure virtual]`

Find char_type matching mask.

This function searches for and returns the first char_type c in [`__lo`,`__hi`) for which `is(__m,c)` is true.

`do_scan_is()` is a hook for a derived facet to change the behavior of match searching. `do_is()` must always return the same result for the same input.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to a matching char_type if found, else __hi.

Implemented in [std::ctype<wchar_t>](#), and [std::ctype<_CharT>](#).

Referenced by [std::__ctype_abstract_base<wchar_t>::narrow\(\)](#), and [std::__ctype_abstract_base<wchar_t>::scan_is\(\)](#).

5.434.3.6 `template<typename _CharT> virtual const char_type* std::__ctype_abstract_base<_CharT>::do_scan_not (mask __m, const char_type * __lo, const char_type * __hi) const` [protected], [pure virtual]

Find char_type not matching mask.

This function searches for and returns a pointer to the first char_type c of [lo,hi) for which is(m,c) is false.

do_scan_is() is a hook for a derived facet to change the behavior of match searching. do_is() must always return the same result for the same input.

Parameters

↩ __m	The mask to compare against.
↩ __lo	Pointer to start of range.
↩ __hi	Pointer to end of range.

Returns

Pointer to a non-matching char_type if found, else __hi.

Implemented in [std::ctype<wchar_t>](#), and [std::ctype<_CharT>](#).

Referenced by [std::__ctype_abstract_base<wchar_t>::narrow\(\)](#), and [std::__ctype_abstract_base<wchar_t>::scan_not\(\)](#).

5.434.3.7 `template<typename _CharT> virtual char_type std::__ctype_abstract_base<_CharT>::do_tolower (char_type __c) const` [protected], [pure virtual]

Convert to lowercase.

This virtual function converts the argument to lowercase if possible. If not possible (for example, '2'), returns the argument.

do_tolower() is a hook for a derived facet to change the behavior of lowercasing. do_tolower() must always return the same result for the same input.

Parameters

\leftrightarrow _c	The char_type to convert.
-------------------------	---------------------------

Returns

The lowercase char_type if convertible, else __c.

Implemented in [std::ctype< wchar_t >](#), and [std::ctype< _CharT >](#).

Referenced by [std::__ctype_abstract_base< wchar_t >::narrow\(\)](#), [std::ctype< char >::table\(\)](#), [std::__ctype_abstract_base< wchar_t >::tolower\(\)](#), and [std::ctype< char >::tolower\(\)](#).

5.434.3.8 `template<typename _CharT> virtual const char_type* std::__ctype_abstract_base< _CharT >::do_tolower (char_type * __lo, const char_type * __hi) const` [protected], [pure virtual]

Convert array to lowercase.

This virtual function converts each char_type in the range [__lo,__hi) to lowercase if possible. Other elements remain untouched.

do_tolower() is a hook for a derived facet to change the behavior of lowercasing. do_tolower() must always return the same result for the same input.

Parameters

\leftrightarrow __lo	Pointer to start of range.
\leftrightarrow __hi	Pointer to end of range.

Returns

__hi.

Implemented in [std::ctype< wchar_t >](#), and [std::ctype< _CharT >](#).

5.434.3.9 `template<typename _CharT> virtual char_type std::__ctype_abstract_base< _CharT >::do_toupper (char_type __c) const` [protected], [pure virtual]

Convert to uppercase.

This virtual function converts the char_type argument to uppercase if possible. If not possible (for example, '2'), returns the argument.

do_toupper() is a hook for a derived facet to change the behavior of uppercasing. do_toupper() must always return the same result for the same input.

Parameters

\leftrightarrow _c	The char_type to convert.
-------------------------	---------------------------

Returns

The uppercase char_type if convertible, else __c.

Implemented in [std::ctype<wchar_t>](#), and [std::ctype<_CharT>](#).

Referenced by [std::__ctype_abstract_base<wchar_t>::narrow\(\)](#), [std::ctype<char>::table\(\)](#), [std::__ctype_abstract_base<wchar_t>::toupper\(\)](#), and [std::ctype<char>::toupper\(\)](#).

5.434.3.10 `template<typename _CharT> virtual const char_type* std::__ctype_abstract_base<_CharT>::do_toupper (char_type * __lo, const char_type * __hi) const` `[protected], [pure virtual]`

Convert array to uppercase.

This virtual function converts each char_type in the range [`__lo`,`__hi`) to uppercase if possible. Other elements remain untouched.

`do_toupper()` is a hook for a derived facet to change the behavior of uppercasing. `do_toupper()` must always return the same result for the same input.

Parameters

\leftrightarrow __lo	Pointer to start of range.
\leftrightarrow __hi	Pointer to end of range.

Returns

`__hi`.

Implemented in [std::ctype<wchar_t>](#), and [std::ctype<_CharT>](#).

5.434.3.11 `template<typename _CharT> virtual char_type std::__ctype_abstract_base<_CharT>::do_widen (char __c) const` `[protected], [pure virtual]`

Widen char.

This virtual function converts the char to char_type using the simplest reasonable transformation.

`do_widen()` is a hook for a derived facet to change the behavior of widening. `do_widen()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>_↔</code>	The char to convert.
<code>_c</code>	

Returns

The converted `char_type`

Implemented in [std::ctype< wchar_t >](#), and [std::ctype< _CharT >](#).

Referenced by `std::__ctype_abstract_base< wchar_t >::narrow()`, `std::__ctype_abstract_base< wchar_t >::widen()`, and `std::ctype< char >::widen()`.

```
5.434.3.12 template<typename _CharT> virtual const char* std::__ctype_abstract_base< _CharT >::do_widen ( const char
* __lo, const char * __hi, char_type * __to ) const [protected], [pure virtual]
```

Widen char array.

This function converts each char in the input to `char_type` using the simplest reasonable transformation.

`do_widen()` is a hook for a derived facet to change the behavior of widening. `do_widen()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>_↔</code> <code>_lo</code>	Pointer to start range.
<code>_↔</code> <code>_hi</code>	Pointer to end of range.
<code>_↔</code> <code>_to</code>	Pointer to the destination array.

Returns

`__hi`.

Implemented in [std::ctype< wchar_t >](#), and [std::ctype< _CharT >](#).

```
5.434.3.13 template<typename _CharT> bool std::__ctype_abstract_base< _CharT >::is ( mask __m, char_type __c )
const [inline]
```

Test `char_type` classification.

This function finds a mask `M` for `__c` and compares it to mask `__m`. It does so by returning the value of `ctype<char_↔ type>::do_is()`.

Parameters

\leftrightarrow _c	The char_type to compare the mask of.
\leftrightarrow _m	The mask to compare against.

Returns

(M & __m) != 0.

Definition at line 169 of file locale_facets.h.

Referenced by std::time_get<_CharT, _InIter>::get(), std::ctype<char>::is(), std::isalnum(), std::isalpha(), std::isblank(), std::iscntrl(), std::isdigit(), std::isgraph(), std::islower(), std::isprint(), std::ispunct(), std::isspace(), std::isupper(), std::isxdigit(), and std::basic_istream<_CharT, _Traits>::sentry::sentry().

5.434.3.14 template<typename _CharT> const char_type* std::__ctype_abstract_base<_CharT>::is (const char_type * __lo, const char_type * __hi, mask * __vec) const [inline]

Return a mask array.

This function finds the mask for each char_type in the range [lo,hi) and successively writes it to vec. vec must have as many elements as the char array. It does so by returning the value of ctype<char_type>::do_is().

Parameters

__lo	Pointer to start of range.
__hi	Pointer to end of range.
__vec	Pointer to an array of mask storage.

Returns

__hi.

Definition at line 186 of file locale_facets.h.

5.434.3.15 template<typename _CharT> char std::__ctype_abstract_base<_CharT>::narrow (char_type __c, char __dfault) const [inline]

Narrow char_type to char.

This function converts the char_type to char using the simplest reasonable transformation. If the conversion fails, dfault is returned instead. It does so by returning ctype<char_type>::do_narrow(__c).

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__c</code>	The <code>char_type</code> to convert.
<code>__default</code>	Char to return if conversion fails.

Returns

The converted char.

Definition at line 331 of file `locale_facets.h`.

Referenced by `std::time_get<_CharT, _InIter>::do_date_order()`, `std::time_get<_CharT, _InIter>::get()`, and `std::time_put<_CharT, _OutIter>::put()`.

```
5.434.3.16 template<typename _CharT> const char_type* std::__ctype_abstract_base<_CharT>::narrow ( const
char_type * __lo, const char_type * __hi, char __default, char * __to ) const [inline]
```

Narrow array to char array.

This function converts each `char_type` in the input to `char` using the simplest reasonable transformation and writes the results to the destination array. For any `char_type` in the input that cannot be converted, `default` is used instead. It does so by returning `ctype<char_type>::do_narrow(__lo, __hi, __default, __to)`.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__default</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Definition at line 353 of file `locale_facets.h`.

```
5.434.3.17 template<typename _CharT> const char_type* std::__ctype_abstract_base<_CharT>::scan_is ( mask __m,
const char_type * __lo, const char_type * __hi ) const [inline]
```

Find `char_type` matching a mask.

This function searches for and returns the first `char_type` `c` in `[lo,hi)` for which `is(m,c)` is true. It does so by returning `ctype<char_type>::do_scan_is()`.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to matching char_type if found, else `__hi`.

Definition at line 202 of file locale_facets.h.

Referenced by `std::ctype<char>::is()`.

5.434.3.18 `template<typename _CharT> const char_type* std::__ctype_abstract_base<_CharT>::scan_not(mask __m, const char_type * __lo, const char_type * __hi) const [inline]`

Find char_type not matching a mask.

This function searches for and returns the first char_type c in [lo,hi) for which is(m,c) is false. It does so by returning `ctype<char_type>::do_scan_not()`.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to first char in range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to non-matching char if found, else `__hi`.

Definition at line 218 of file locale_facets.h.

Referenced by `std::money_get<_CharT, _InIter>::do_get()`, and `std::ctype<char>::scan_is()`.

5.434.3.19 `template<typename _CharT> char_type std::__ctype_abstract_base<_CharT>::tolower(char_type __c) const [inline]`

Convert to lowercase.

This function converts the argument to lowercase if possible. If not possible (for example, '2'), returns the argument. It does so by returning `ctype<char_type>::do_tolower(c)`.

Parameters

<code>__c</code>	The char_type to convert.
------------------	---------------------------

Returns

The lowercase char_type if convertible, else `__c`.

Definition at line 261 of file locale_facets.h.

Referenced by `std::time_get<_CharT, _InIter >::get()`, and `std::tolower()`.

5.434.3.20 `template<typename _CharT> const char_type* std::__ctype_abstract_base<_CharT>::tolower (char_type * __lo, const char_type * __hi) const [inline]`

Convert array to lowercase.

This function converts each char_type in the range `[__lo,__hi)` to lowercase if possible. Other elements remain untouched. It does so by returning `ctype<char_type>::do_tolower(__lo, __hi)`.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

Definition at line 276 of file locale_facets.h.

5.434.3.21 `template<typename _CharT> char_type std::__ctype_abstract_base<_CharT>::toupper (char_type __c) const [inline]`

Convert to uppercase.

This function converts the argument to uppercase if possible. If not possible (for example, '2'), returns the argument. It does so by returning `ctype<char_type>::do_toupper()`.

Parameters

<code>__c</code>	The char_type to convert.
------------------	---------------------------

Returns

The uppercase char_type if convertible, else __c.

Definition at line 232 of file locale_facets.h.

Referenced by std::time_get<_CharT, _InIter>::do_date_order(), std::time_get<_CharT, _InIter>::get(), and std::__toupper().

5.434.3.22 `template<typename _CharT> const char_type* std::__ctype_abstract_base<_CharT>::toupper (char_type * __lo, const char_type * __hi) const [inline]`

Convert array to uppercase.

This function converts each char_type in the range [lo,hi) to uppercase if possible. Other elements remain untouched. It does so by returning ctype<char_type>::do_toupper(lo, hi).

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

Definition at line 247 of file locale_facets.h.

5.434.3.23 `template<typename _CharT> char_type std::__ctype_abstract_base<_CharT>::widen (char __c) const [inline]`

Widen char to char_type.

This function converts the char argument to char_type using the simplest reasonable transformation. It does so by returning ctype<char_type>::do_widen(c).

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__c</code>	The char to convert.
------------------	----------------------

Returns

The converted char_type.

Definition at line 293 of file locale_facets.h.

Referenced by `std::time_get< _CharT, _Inlter >::do_get()`, `std::money_get< _CharT, _Inlter >::do_get()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::time_put< _CharT, _Outlter >::do_put()`, `std::money_put< _CharT, _Outlter >::do_put()`, and `std::num_put< _CharT, _Outlter >::do_put()`.

5.434.3.24 `template<typename _CharT> const char* std::__ctype_abstract_base< _CharT >::widen (const char * __lo, const char * __hi, char_type * __to) const [inline]`

Widen array to `char_type`.

This function converts each char in the input to `char_type` using the simplest reasonable transformation. It does so by returning `ctype<char_type>::do_widen(c)`.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Definition at line 312 of file locale_facets.h.

The documentation for this class was generated from the following file:

- [locale_facets.h](#)

5.435 `std::__debug::bitset< _Nb >` Class Template Reference

Inherits `bitset< _Nb >`.

Public Types

- `typedef _Base::reference` **reference**

Public Member Functions

- constexpr **bitset** (unsigned long long __val) noexcept
- template<typename _CharT, typename _Traits, typename _Alloc >
bitset (const [std::basic_string](#)< _CharT, _Traits, _Alloc > &__str, typename [std::basic_string](#)< _CharT, _Traits, _Alloc >::size_type __pos=0, typename [std::basic_string](#)< _CharT, _Traits, _Alloc >::size_type __n=[std::basic_string](#)< _CharT, _Traits, _Alloc >::npos))
- template<class _CharT, class _Traits, class _Alloc >
bitset (const [std::basic_string](#)< _CharT, _Traits, _Alloc > &__str, typename [std::basic_string](#)< _CharT, _Traits, _Alloc >::size_type __pos, typename [std::basic_string](#)< _CharT, _Traits, _Alloc >::size_type __n, _CharT __zero, _CharT __one=_CharT('1'))
- **bitset** (const [_Base](#) &__x)
- template<typename _CharT >
bitset (const _CharT *__str, typename [std::basic_string](#)< _CharT >::size_type __n=[std::basic_string](#)< _CharT >::npos, _CharT __zero=_CharT('0'), _CharT __one=_CharT('1'))
- [_Base](#) & [_M_base](#) () noexcept
- const [_Base](#) & [_M_base](#) () const noexcept
- [bitset](#)< _Nb > & **flip** () noexcept
- [bitset](#)< _Nb > & **flip** (size_t __pos)
- bool **operator!=** (const [bitset](#)< _Nb > &__rhs) const noexcept
- [bitset](#)< _Nb > & **operator&=** (const [bitset](#)< _Nb > &__rhs) noexcept
- [bitset](#)< _Nb > **operator<<** (size_t __pos) const noexcept
- [bitset](#)< _Nb > & **operator<<=** (size_t __pos) noexcept
- bool **operator==** (const [bitset](#)< _Nb > &__rhs) const noexcept
- [bitset](#)< _Nb > **operator>>** (size_t __pos) const noexcept
- [bitset](#)< _Nb > & **operator>>=** (size_t __pos) noexcept
- reference **operator[]** (size_t __pos)
- constexpr bool **operator[]** (size_t __pos) const
- [bitset](#)< _Nb > & **operator^=** (const [bitset](#)< _Nb > &__rhs) noexcept
- [bitset](#)< _Nb > & **operator|=** (const [bitset](#)< _Nb > &__rhs) noexcept
- [bitset](#)< _Nb > **operator~** () const noexcept
- [bitset](#)< _Nb > & **reset** () noexcept
- [bitset](#)< _Nb > & **reset** (size_t __pos)
- [bitset](#)< _Nb > & **set** () noexcept
- [bitset](#)< _Nb > & **set** (size_t __pos, bool __val=true)
- template<typename _CharT, typename _Traits, typename _Alloc >
[std::basic_string](#)< _CharT, _Traits, _Alloc > **to_string** () const
- template<class _CharT, class _Traits, class _Alloc >
[std::basic_string](#)< _CharT, _Traits, _Alloc > **to_string** (_CharT __zero, _CharT __one=_CharT('1')) const
- template<typename _CharT, typename _Traits >
[std::basic_string](#)< _CharT, _Traits, [std::allocator](#)< _CharT > > **to_string** () const
- template<class _CharT, class _Traits >
[std::basic_string](#)< _CharT, _Traits, [std::allocator](#)< _CharT > > **to_string** (_CharT __zero, _CharT __one=_CharT('1')) const
- template<typename _CharT >
[std::basic_string](#)< _CharT, [std::char_traits](#)< _CharT >, [std::allocator](#)< _CharT > > **to_string** () const
- template<class _CharT >
[std::basic_string](#)< _CharT, [std::char_traits](#)< _CharT >, [std::allocator](#)< _CharT > > **to_string** (_CharT __zero, _CharT __one=_CharT('1')) const
- [std::basic_string](#)< char, [std::char_traits](#)< char >, [std::allocator](#)< char > > **to_string** () const
- [std::basic_string](#)< char, [std::char_traits](#)< char >, [std::allocator](#)< char > > **to_string** (char __zero, char __one='1') const

5.435.1 Detailed Description

```
template<size_t _Nb>
class std::__debug::bitset< _Nb >
```

Class std::bitset with additional safety/checking/debug instrumentation.

Definition at line 42 of file debug/bitset.

The documentation for this class was generated from the following file:

- [debug/bitset](#)

5.436 std::__debug::deque< _Tp, _Allocator > Class Template Reference

Inheritance diagram for std::__debug::deque< _Tp, _Allocator >:



Public Types

- typedef _Allocator **allocator_type**
- typedef [__gnu_debug::__Safe_iterator](#)< [_Base_const_iterator](#), [deque](#) > **const_iterator**
- typedef _Base::const_pointer **const_pointer**
- typedef _Base::const_reference **const_reference**
- typedef [std::reverse_iterator](#)< [const_iterator](#) > **const_reverse_iterator**
- typedef _Base::difference_type **difference_type**
- typedef [__gnu_debug::__Safe_iterator](#)< [_Base_iterator](#), [deque](#) > **iterator**
- typedef _Base::pointer **pointer**
- typedef _Base::reference **reference**
- typedef [std::reverse_iterator](#)< [iterator](#) > **reverse_iterator**
- typedef _Base::size_type **size_type**
- typedef _Tp **value_type**

Public Member Functions

- **deque** (const [deque](#) &)=default
- **deque** ([deque](#) &&)=default
- **deque** (const [deque](#) &__d, const _Allocator &__a)
- **deque** ([deque](#) &&__d, const _Allocator &__a)
- **deque** ([initializer_list](#)< value_type > __l, const allocator_type &__a=allocator_type())
- **deque** (const _Allocator &__a)
- **deque** (size_type __n, const _Allocator &__a=_Allocator())
- **deque** (size_type __n, const _Tp &__value, const _Allocator &__a=_Allocator())
- template<class _InputIterator, typename = std::RequireInputIter<_InputIterator>>>
 deque (_InputIterator __first, _InputIterator __last, const _Allocator &__a=_Allocator())
- **deque** (const [_Base](#) &__x)
- void **_M_attach** (_Safe_iterator_base *__it, bool __constant)
- void **_M_attach_single** (_Safe_iterator_base *__it, bool __constant) throw ()
- [_Base](#) & **_M_base** () noexcept
- const [_Base](#) & **_M_base** () const noexcept
- void **_M_detach** (_Safe_iterator_base *__it)
- void **_M_detach_single** (_Safe_iterator_base *__it) throw ()
- void **_M_invalidate_all** () const
- void **_M_invalidate_if** (_Predicate __pred)
- void **_M_swap** (_Safe_container &__x) noexcept
- void **_M_transfer_from_if** (_Safe_sequence &__from, _Predicate __pred)
- template<class _InputIterator, typename = std::RequireInputIter<_InputIterator>>>
 void **assign** (_InputIterator __first, _InputIterator __last)
- void **assign** (size_type __n, const _Tp &__t)
- void **assign** ([initializer_list](#)< value_type > __l)
- reference **back** () noexcept
- const_reference **back** () const noexcept
- [iterator](#) **begin** () noexcept
- const_iterator **begin** () const noexcept
- const_iterator **cbegin** () const noexcept
- const_iterator **cend** () const noexcept
- void **clear** () noexcept
- const_reverse_iterator **crbegin** () const noexcept
- const_reverse_iterator **crend** () const noexcept
- template<typename... _Args>
 [iterator](#) **emplace** (const_iterator __position, _Args &&...__args)
- template<typename... _Args>
 void **emplace_back** (_Args &&...__args)
- template<typename... _Args>
 void **emplace_front** (_Args &&...__args)
- [iterator](#) **end** () noexcept
- const_iterator **end** () const noexcept
- [iterator](#) **erase** (const_iterator __position)
- [iterator](#) **erase** (const_iterator __first, const_iterator __last)
- reference **front** () noexcept
- const_reference **front** () const noexcept
- [iterator](#) **insert** (const_iterator __position, const _Tp &__x)
- [iterator](#) **insert** (const_iterator __position, _Tp &&__x)
- [iterator](#) **insert** (const_iterator __position, [initializer_list](#)< value_type > __l)

- **iterator insert** ([const_iterator](#) __position, size_type __n, const _Tp &__x)
- template<class _InputIterator, typename = std::enable_if_t<is_input_iterator<_InputIterator>::value>>
iterator insert ([const_iterator](#) __position, _InputIterator __first, _InputIterator __last)
- **deque & operator=** (const [deque](#) &)=default
- **deque & operator=** ([deque](#) &&)=default
- **deque & operator=** ([initializer_list](#)< value_type > __l)
- reference **operator[]** (size_type __n) noexcept
- const_reference **operator[]** (size_type __n) const noexcept
- void **pop_back** () noexcept
- void **pop_front** () noexcept
- void **push_back** (const _Tp &__x)
- void **push_back** (_Tp &&__x)
- void **push_front** (const _Tp &__x)
- void **push_front** (_Tp &&__x)
- [reverse_iterator](#) **rbegin** () noexcept
- [const_reverse_iterator](#) **rbegin** () const noexcept
- [reverse_iterator](#) **rend** () noexcept
- [const_reverse_iterator](#) **rend** () const noexcept
- void **resize** (size_type __sz)
- void **resize** (size_type __sz, const _Tp &__c)
- void **shrink_to_fit** () noexcept
- void **swap** ([deque](#) &__x) noexcept(*conditional *)

Public Attributes

- _Safe_iterator_base * [_M_const_iterators](#)
- _Safe_iterator_base * [_M_iterators](#)
- unsigned int [_M_version](#)

Protected Member Functions

- void [_M_detach_all](#) ()
- void [_M_detach_singular](#) ()
- __gnu_cxx::__mutex & [_M_get_mutex](#) () throw ()
- void [_M_revalidate_singular](#) ()
- _Safe_container & [_M_safe](#) () noexcept
- void [_M_swap](#) (_Safe_sequence_base &__x) noexcept

5.436.1 Detailed Description

```
template<typename _Tp, typename _Allocator = std::allocator<_Tp>>
class std::__debug::deque<_Tp, _Allocator>
```

Class std::deque with safety/checking/debug instrumentation.

Definition at line 43 of file debug/deque.

5.436.2 Member Function Documentation

5.436.2.1 void __gnu_debug::Safe_sequence_base::M_attach (_Safe_iterator_base * __it, bool __constant)
[inherited]

Attach an iterator to this sequence.

5.436.2.2 void __gnu_debug::Safe_sequence_base::M_attach_single (_Safe_iterator_base * __it, bool __constant) throw)
[inherited]

Likewise but not thread safe.

5.436.2.3 void __gnu_debug::Safe_sequence_base::M_detach (_Safe_iterator_base * __it) [inherited]

Detach an iterator from this sequence

Referenced by __gnu_debug::Safe_iterator< _Iterator, _Sequence >::Safe_iterator().

5.436.2.4 void __gnu_debug::Safe_sequence_base::M_detach_all () [protected],[inherited]

Detach all iterators, leaving them singular.

5.436.2.5 void __gnu_debug::Safe_sequence_base::M_detach_single (_Safe_iterator_base * __it) throw)
[inherited]

Likewise but not thread safe.

5.436.2.6 void __gnu_debug::Safe_sequence_base::M_detach_singular () [protected],[inherited]

Detach all singular iterators.

Postcondition

for all iterators i attached to this sequence, i->_M_version == _M_version.

5.436.2.7 __gnu_cxx::__mutex& __gnu_debug::Safe_sequence_base::M_get_mutex () throw) [protected],
[inherited]

For use in _Safe_sequence.

Referenced by __gnu_debug::Safe_sequence< _Sequence >::M_transfer_from_if().

5.436.2.8 void __gnu_debug::Safe_sequence_base::M_invalidate_all () const [inline],[inherited]

Invalidates all iterators.

Definition at line 248 of file safe_base.h.

References __gnu_debug::Safe_iterator_base::M_attach(), __gnu_debug::Safe_iterator_base::M_attach_single(), __gnu_debug::Safe_iterator_base::M_detach(), and __gnu_debug::Safe_iterator_base::M_detach_single().

5.436.2.9 void `__gnu_debug::Safe_sequence< deque< _Tp, _Allocator > >::M_invalidate_if (_Predicate __pred)` `[inherited]`

Invalidates all iterators `x` that reference this sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

5.436.2.10 void `__gnu_debug::Safe_sequence_base::M_revalidate_singular ()` `[protected]`, `[inherited]`

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

5.436.2.11 void `__gnu_debug::Safe_sequence_base::M_swap (_Safe_sequence_base & __x)` `[protected]`, `[noexcept]`, `[inherited]`

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.436.2.12 void `__gnu_debug::Safe_sequence< deque< _Tp, _Allocator > >::M_transfer_from_if (_Safe_sequence< deque< _Tp, _Allocator > > & __from, _Predicate __pred)` `[inherited]`

Transfers all iterators `x` that reference `from` sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

5.436.3 Member Data Documentation

5.436.3.1 `_Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_const_iterators` `[inherited]`

The list of constant iterators that reference this container.

Definition at line 193 of file `safe_base.h`.

Referenced by `__gnu_debug::Safe_sequence< _Sequence >::M_transfer_from_if()`.

5.436.3.2 `_Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_iterators` `[inherited]`

The list of mutable iterators that reference this container.

Definition at line 190 of file `safe_base.h`.

Referenced by `__gnu_debug::Safe_sequence< _Sequence >::M_transfer_from_if()`.

5.436.3.3 `unsigned int __gnu_debug::Safe_sequence_base::M_version` `[mutable]`, `[inherited]`

The container version number. This number may never be 0.

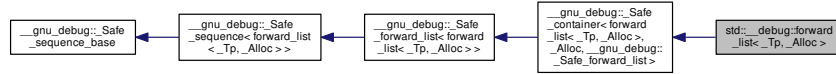
Definition at line 196 of file `safe_base.h`.

The documentation for this class was generated from the following file:

- [debug/deque](#)

5.437 std::__debug::forward_list< _Tp, _Alloc > Class Template Reference

Inheritance diagram for std::__debug::forward_list< _Tp, _Alloc >:



Public Types

- typedef `_Base::allocator_type` **allocator_type**
- typedef `__gnu_debug::__Safe_iterator< _Base_const_iterator, forward_list >` **const_iterator**
- typedef `_Base::const_pointer` **const_pointer**
- typedef `_Base::const_reference` **const_reference**
- typedef `_Base::difference_type` **difference_type**
- typedef `__gnu_debug::__Safe_iterator< _Base_iterator, forward_list >` **iterator**
- typedef `_Base::pointer` **pointer**
- typedef `_Base::reference` **reference**
- typedef `_Base::size_type` **size_type**
- typedef `_Tp` **value_type**

Public Member Functions

- **forward_list** (const allocator_type &__al) noexcept
- **forward_list** (const forward_list &__list, const allocator_type &__al)
- **forward_list** (forward_list && __list, const allocator_type &__al)
- **forward_list** (size_type __n, const allocator_type &__al=allocator_type())
- **forward_list** (size_type __n, const _Tp &__value, const allocator_type &__al=allocator_type())
- template<typename _InputIterator, typename = std::RequireInputIter<_InputIterator>>>
 forward_list (_InputIterator __first, _InputIterator __last, const allocator_type &__al=allocator_type())
- **forward_list** (const forward_list &)=default
- **forward_list** (forward_list &&)=default
- **forward_list** (std::initializer_list< _Tp > __il, const allocator_type &__al=allocator_type())
- void **_M_attach** (_Safe_iterator_base * __it, bool __constant)
- void **_M_attach_single** (_Safe_iterator_base * __it, bool __constant) throw ()
- **_Base & _M_base** () noexcept
- const **_Base & _M_base** () const noexcept
- void **_M_detach** (_Safe_iterator_base * __it)
- void **_M_detach_single** (_Safe_iterator_base * __it) throw ()
- void **_M_invalidate_all** () const
- void **_M_invalidate_if** (_Predicate __pred)
- void **_M_swap** (_Safe_container & __x) noexcept
- void **_M_transfer_from_if** (_Safe_sequence & __from, _Predicate __pred)
- template<typename _InputIterator, typename = std::RequireInputIter<_InputIterator>>>
 void **assign** (_InputIterator __first, _InputIterator __last)
- void **assign** (size_type __n, const _Tp & __val)

- void **assign** (std::initializer_list< _Tp > __il)
- **iterator before_begin** () noexcept
- **const_iterator before_begin** () const noexcept
- **iterator begin** () noexcept
- **const_iterator begin** () const noexcept
- **const_iterator cbefore_begin** () const noexcept
- **const_iterator cbegin** () const noexcept
- **const_iterator cend** () const noexcept
- void **clear** () noexcept
- template<typename... _Args>
 iterator emplace_after (const_iterator __pos, _Args &&... __args)
- **iterator end** () noexcept
- **const_iterator end** () const noexcept
- **iterator erase_after** (const_iterator __pos)
- **iterator erase_after** (const_iterator __pos, const_iterator __last)
- reference **front** ()
- const_reference **front** () const
- **iterator insert_after** (const_iterator __pos, const _Tp & __val)
- **iterator insert_after** (const_iterator __pos, _Tp && __val)
- **iterator insert_after** (const_iterator __pos, size_type __n, const _Tp & __val)
- template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>>
 iterator insert_after (const_iterator __pos, _InputIterator __first, _InputIterator __last)
- **iterator insert_after** (const_iterator __pos, std::initializer_list< _Tp > __il)
- void **merge** (forward_list && __list)
- void **merge** (forward_list & __list)
- template<typename _Comp >
 void **merge** (forward_list && __list, _Comp __comp)
- template<typename _Comp >
 void **merge** (forward_list & __list, _Comp __comp)
- **forward_list & operator=** (const forward_list &)=default
- **forward_list & operator=** (forward_list &&)=default
- **forward_list & operator=** (std::initializer_list< _Tp > __il)
- void **pop_front** ()
- void **remove** (const _Tp & __val)
- template<typename _Pred >
 void **remove_if** (_Pred __pred)
- void **resize** (size_type __sz)
- void **resize** (size_type __sz, const value_type & __val)
- void **splice_after** (const_iterator __pos, forward_list && __list)
- void **splice_after** (const_iterator __pos, forward_list & __list)
- void **splice_after** (const_iterator __pos, forward_list && __list, const_iterator __i)
- void **splice_after** (const_iterator __pos, forward_list & __list, const_iterator __i)
- void **splice_after** (const_iterator __pos, forward_list && __list, const_iterator __before, const_iterator __last)
- void **splice_after** (const_iterator __pos, forward_list & __list, const_iterator __before, const_iterator __last)
- void **swap** (forward_list & __list) noexcept(noexcept(declval< _Base & >().swap(__list)))
- void **unique** ()
- template<typename _BinPred >
 void **unique** (_BinPred __binary_pred)

Public Attributes

- [_Safe_iterator_base](#) * [_M_const_iterators](#)
- [_Safe_iterator_base](#) * [_M_iterators](#)
- unsigned int [_M_version](#)

Protected Member Functions

- void [_M_detach_all](#) ()
- void [_M_detach_singular](#) ()
- [__gnu_cxx::__mutex](#) & [_M_get_mutex](#) () throw ()
- void [_M_invalidate_all](#) ()
- void [_M_revalidate_singular](#) ()
- [_Safe_container](#) & [_M_safe](#) () noexcept
- void [_M_swap](#) ([_Safe_sequence_base](#) &) noexcept

5.437.1 Detailed Description

```
template<typename _Tp, typename _Alloc = std::allocator<_Tp>>
class std::__debug::forward_list<_Tp, _Alloc>
```

Class std::forward_list with safety/checking/debug instrumentation.

Definition at line 184 of file debug/forward_list.

5.437.2 Member Function Documentation

5.437.2.1 void [__gnu_debug::Safe_sequence_base::M_attach](#) ([_Safe_iterator_base](#) * [__it](#), bool [__constant](#))
[inherited]

Attach an iterator to this sequence.

5.437.2.2 void [__gnu_debug::Safe_sequence_base::M_attach_single](#) ([_Safe_iterator_base](#) * [__it](#), bool [__constant](#)) throw()
[inherited]

Likewise but not thread safe.

5.437.2.3 void [__gnu_debug::Safe_sequence_base::M_detach](#) ([_Safe_iterator_base](#) * [__it](#)) [inherited]

Detach an iterator from this sequence

Referenced by [__gnu_debug::Safe_iterator<_Iterator, _Sequence>::Safe_iterator\(\)](#).

5.437.2.4 void [__gnu_debug::Safe_sequence_base::M_detach_all](#) () [protected], [inherited]

Detach all iterators, leaving them singular.

5.437.2.5 void __gnu_debug::Safe_sequence_base::M_detach_single (_Safe_iterator_base * __it) throw)
[inherited]

Likewise but not thread safe.

5.437.2.6 void __gnu_debug::Safe_sequence_base::M_detach_singular () [protected],[inherited]

Detach all singular iterators.

Postcondition

for all iterators i attached to this sequence, i->_M_version == _M_version.

5.437.2.7 __gnu_cxx::mutex& __gnu_debug::Safe_sequence_base::M_get_mutex () throw) [protected],
[inherited]

For use in _Safe_sequence.

Referenced by __gnu_debug::Safe_sequence< _Sequence >::M_transfer_from_if().

5.437.2.8 void __gnu_debug::Safe_sequence_base::M_invalidate_all () const [inline],[inherited]

Invalidates all iterators.

Definition at line 248 of file safe_base.h.

References __gnu_debug::Safe_iterator_base::M_attach(), __gnu_debug::Safe_iterator_base::M_attach_single(), __gnu_debug::Safe_iterator_base::M_detach(), and __gnu_debug::Safe_iterator_base::M_detach_single().

5.437.2.9 void __gnu_debug::Safe_sequence< forward_list< _Tp, _Alloc > >::M_invalidate_if (_Predicate __pred)
[inherited]

Invalidates all iterators x that reference this sequence, are not singular, and for which __pred(x) returns true. __pred will be invoked with the normal iterators nested in the safe ones.

5.437.2.10 void __gnu_debug::Safe_sequence_base::M_revalidate_singular () [protected],[inherited]

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

5.437.2.11 void __gnu_debug::Safe_sequence< forward_list< _Tp, _Alloc > >::M_transfer_from_if (_Safe_sequence< forward_list< _Tp, _Alloc > > & __from, _Predicate __pred) [inherited]

Transfers all iterators x that reference from sequence, are not singular, and for which __pred(x) returns true. __pred will be invoked with the normal iterators nested in the safe ones.

5.437.3 Member Data Documentation

5.437.3.1 `_Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_const_iterators` [inherited]

The list of constant iterators that reference this container.

Definition at line 193 of file `safe_base.h`.

Referenced by `__gnu_debug::Safe_sequence<_Sequence>::_M_transfer_from_if()`.

5.437.3.2 `_Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_iterators` [inherited]

The list of mutable iterators that reference this container.

Definition at line 190 of file `safe_base.h`.

Referenced by `__gnu_debug::Safe_sequence<_Sequence>::_M_transfer_from_if()`.

5.437.3.3 `unsigned int __gnu_debug::Safe_sequence_base::M_version` [mutable], [inherited]

The container version number. This number may never be 0.

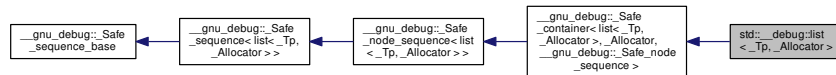
Definition at line 196 of file `safe_base.h`.

The documentation for this class was generated from the following file:

- [debug/forward_list](#)

5.438 `std::__debug::list<_Tp, _Allocator>` Class Template Reference

Inheritance diagram for `std::__debug::list<_Tp, _Allocator>`:



Public Types

- typedef `_Allocator` **allocator_type**
- typedef `__gnu_debug::Safe_iterator<_Base_const_iterator, list>` **const_iterator**
- typedef `_Base::const_pointer` **const_pointer**
- typedef `_Base::const_reference` **const_reference**
- typedef `std::reverse_iterator<const_iterator>` **const_reverse_iterator**
- typedef `_Base::difference_type` **difference_type**
- typedef `__gnu_debug::Safe_iterator<_Base_iterator, list>` **iterator**
- typedef `_Base::pointer` **pointer**
- typedef `_Base::reference` **reference**
- typedef `std::reverse_iterator<iterator>` **reverse_iterator**
- typedef `_Base::size_type` **size_type**
- typedef `_Tp` **value_type**

Public Member Functions

- **list** (const [list](#) &)=default
- **list** ([list](#) &&)=default
- **list** ([initializer_list](#)< value_type > __l, const allocator_type &__a=allocator_type())
- **list** (const [list](#) &__x, const allocator_type &__a)
- **list** ([list](#) &&__x, const allocator_type &__a)
- **list** (const _Allocator &__a) noexcept
- **list** (size_type __n, const allocator_type &__a=allocator_type())
- **list** (size_type __n, const _Tp &__value, const _Allocator &__a=_Allocator())
- template<class _InputIterator, typename = std::::RequireInputIter<_InputIterator>>>
 list (_InputIterator __first, _InputIterator __last, const _Allocator &__a=_Allocator())
- **list** (const [_Base](#) &__x)
- void [_M_attach](#) (_Safe_iterator_base *__it, bool __constant)
- void [_M_attach_single](#) (_Safe_iterator_base *__it, bool __constant) throw ()
- [_Base](#) & [_M_base](#) () noexcept
- const [_Base](#) & [_M_base](#) () const noexcept
- void [_M_detach](#) (_Safe_iterator_base *__it)
- void [_M_detach_single](#) (_Safe_iterator_base *__it) throw ()
- void [_M_invalidate_all](#) () const
- void [_M_invalidate_if](#) (_Predicate __pred)
- void [_M_swap](#) (_Safe_container &__x) noexcept
- void [_M_transfer_from_if](#) (_Safe_sequence &__from, _Predicate __pred)
- void **assign** ([initializer_list](#)< value_type > __l)
- template<class _InputIterator, typename = std::::RequireInputIter<_InputIterator>>>
 void **assign** (_InputIterator __first, _InputIterator __last)
- void **assign** (size_type __n, const _Tp &__t)
- reference **back** () noexcept
- const_reference **back** () const noexcept
- iterator **begin** () noexcept
- const_iterator **begin** () const noexcept
- const_iterator **cbegin** () const noexcept
- const_iterator **cend** () const noexcept
- void **clear** () noexcept
- const_reverse_iterator **crbegin** () const noexcept
- const_reverse_iterator **crend** () const noexcept
- template<typename... _Args>
 iterator **emplace** (const_iterator __position, _Args &&...__args)
- iterator **end** () noexcept
- const_iterator **end** () const noexcept
- iterator **erase** (const_iterator __position) noexcept
- iterator **erase** (const_iterator __first, const_iterator __last) noexcept
- reference **front** () noexcept
- const_reference **front** () const noexcept
- iterator **insert** (const_iterator __position, const _Tp &__x)
- iterator **insert** (const_iterator __position, _Tp &&__x)
- iterator **insert** (const_iterator __p, [initializer_list](#)< value_type > __l)
- iterator **insert** (const_iterator __position, size_type __n, const _Tp &__x)
- template<class _InputIterator, typename = std::::RequireInputIter<_InputIterator>>>
 iterator **insert** (const_iterator __position, _InputIterator __first, _InputIterator __last)
- void **merge** ([list](#) &&__x)

- void **merge** (list &__x)
- template<class _Compare >
void **merge** (list &&__x, _Compare __comp)
- template<typename _Compare >
void **merge** (list &__x, _Compare __comp)
- list & **operator=** (const list &)=default
- list & **operator=** (list &&)=default
- list & **operator=** (initializer_list< value_type > __l)
- void **pop_back** () noexcept
- void **pop_front** () noexcept
- **reverse_iterator** **rbegin** () noexcept
- **const_reverse_iterator** **rbegin** () const noexcept
- void **remove** (const _Tp &__value)
- template<class _Predicate >
void **remove_if** (_Predicate __pred)
- **reverse_iterator** **rend** () noexcept
- **const_reverse_iterator** **rend** () const noexcept
- void **resize** (size_type __sz)
- void **resize** (size_type __sz, const _Tp &__c)
- void **sort** ()
- template<typename _StrictWeakOrdering >
void **sort** (_StrictWeakOrdering __pred)
- void **splice** (const_iterator __position, list &&__x) noexcept
- void **splice** (const_iterator __position, list &__x) noexcept
- void **splice** (const_iterator __position, list &&__x, const_iterator __i) noexcept
- void **splice** (const_iterator __position, list &__x, const_iterator __i) noexcept
- void **splice** (const_iterator __position, list &&__x, const_iterator __first, const_iterator __last) noexcept
- void **splice** (const_iterator __position, list &__x, const_iterator __first, const_iterator __last) noexcept
- void **swap** (list &__x) noexcept(/*conditional */)
- void **unique** ()
- template<class _BinaryPredicate >
void **unique** (_BinaryPredicate __binary_pred)

Public Attributes

- _Safe_iterator_base * **_M_const_iterators**
- _Safe_iterator_base * **_M_iterators**
- unsigned int **_M_version**

Protected Member Functions

- void **_M_detach_all** ()
- void **_M_detach_singular** ()
- __gnu_cxx::__mutex & **_M_get_mutex** () throw ()
- void **_M_invalidate_all** ()
- void **_M_revalidate_singular** ()
- _Safe_container & **_M_safe** () noexcept
- void **_M_swap** (_Safe_sequence_base &__x) noexcept

5.438.1 Detailed Description

```
template<typename _Tp, typename _Allocator = std::allocator<_Tp>>
class std::__debug::list< _Tp, _Allocator >
```

Class std::list with safety/checking/debug instrumentation.

Definition at line 43 of file debug/list.

5.438.2 Member Function Documentation

```
5.438.2.1 void __gnu_debug::Safe_sequence_base::M_attach ( _Safe_iterator_base * __it, bool __constant )
[inherited]
```

Attach an iterator to this sequence.

```
5.438.2.2 void __gnu_debug::Safe_sequence_base::M_attach_single ( _Safe_iterator_base * __it, bool __constant ) throw )
[inherited]
```

Likewise but not thread safe.

```
5.438.2.3 void __gnu_debug::Safe_sequence_base::M_detach ( _Safe_iterator_base * __it ) [inherited]
```

Detach an iterator from this sequence

Referenced by __gnu_debug::Safe_iterator< _Iterator, _Sequence >::Safe_iterator().

```
5.438.2.4 void __gnu_debug::Safe_sequence_base::M_detach_all ( ) [protected],[inherited]
```

Detach all iterators, leaving them singular.

```
5.438.2.5 void __gnu_debug::Safe_sequence_base::M_detach_single ( _Safe_iterator_base * __it ) throw )
[inherited]
```

Likewise but not thread safe.

```
5.438.2.6 void __gnu_debug::Safe_sequence_base::M_detach_singular ( ) [protected],[inherited]
```

Detach all singular iterators.

Postcondition

for all iterators *i* attached to this sequence, *i*->_M_version == _M_version.

5.438.2.7 `__gnu_cxx::mutex& __gnu_debug::Safe_sequence_base::M_get_mutex () throw` [protected],
[inherited]

For use in `_Safe_sequence`.

Referenced by `__gnu_debug::Safe_sequence<_Sequence >::M_transfer_from_if()`.

5.438.2.8 `void __gnu_debug::Safe_sequence_base::M_invalidate_all () const` [inline], [inherited]

Invalidates all iterators.

Definition at line 248 of file `safe_base.h`.

References `__gnu_debug::Safe_iterator_base::M_attach()`, `__gnu_debug::Safe_iterator_base::M_attach_single()`, `__gnu_debug::Safe_iterator_base::M_detach()`, and `__gnu_debug::Safe_iterator_base::M_detach_single()`.

5.438.2.9 `void __gnu_debug::Safe_sequence<list<_Tp, _Allocator > >::M_invalidate_if (_Predicate __pred)`
[inherited]

Invalidates all iterators `x` that reference this sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

5.438.2.10 `void __gnu_debug::Safe_sequence_base::M_revalidate_singular ()` [protected], [inherited]

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

5.438.2.11 `void __gnu_debug::Safe_sequence_base::M_swap (_Safe_sequence_base & __x)` [protected],
[noexcept], [inherited]

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.438.2.12 `void __gnu_debug::Safe_sequence<list<_Tp, _Allocator > >::M_transfer_from_if (_Safe_sequence<list<_Tp, _Allocator > > & __from, _Predicate __pred)` [inherited]

Transfers all iterators `x` that reference `from` sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

5.438.3 Member Data Documentation

5.438.3.1 `_Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_const_iterators` [inherited]

The list of constant iterators that reference this container.

Definition at line 193 of file `safe_base.h`.

Referenced by `__gnu_debug::Safe_sequence<_Sequence >::M_transfer_from_if()`.

5.438.3.2 `_Safe_iterator_base* __gnu_debug::_Safe_sequence_base::_M_iterators` [inherited]

The list of mutable iterators that reference this container.

Definition at line 190 of file `safe_base.h`.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`.

5.438.3.3 `unsigned int __gnu_debug::_Safe_sequence_base::_M_version` [mutable], [inherited]

The container version number. This number may never be 0.

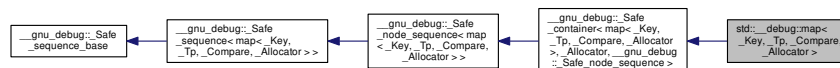
Definition at line 196 of file `safe_base.h`.

The documentation for this class was generated from the following file:

- [debug/list](#)

5.439 `std::__debug::map<_Key, _Tp, _Compare, _Allocator>` Class Template Reference

Inheritance diagram for `std::__debug::map<_Key, _Tp, _Compare, _Allocator>`:



Public Types

- typedef `_Allocator` **allocator_type**
- typedef `__gnu_debug::_Safe_iterator<_Base_const_iterator, map>` **const_iterator**
- typedef `_Base::const_pointer` **const_pointer**
- typedef `_Base::const_reference` **const_reference**
- typedef `std::reverse_iterator<const_iterator>` **const_reverse_iterator**
- typedef `_Base::difference_type` **difference_type**
- typedef `__gnu_debug::_Safe_iterator<_Base_iterator, map>` **iterator**
- typedef `_Compare` **key_compare**
- typedef `_Key` **key_type**
- typedef `_Tp` **mapped_type**
- typedef `_Base::pointer` **pointer**
- typedef `_Base::reference` **reference**
- typedef `std::reverse_iterator<iterator>` **reverse_iterator**
- typedef `_Base::size_type` **size_type**
- typedef `std::pair<const _Key, _Tp>` **value_type**

Public Member Functions

- **map** (const [map](#) &)=default
- **map** ([map](#) &&)=default
- **map** ([initializer_list](#)< [value_type](#) > __l, const _Compare &__c=_Compare(), const allocator_type &__a=allocator_type())
- **map** (const allocator_type &__a)
- **map** (const [map](#) &__m, const allocator_type &__a)
- **map** ([map](#) &&__m, const allocator_type &__a)
- **map** ([initializer_list](#)< [value_type](#) > __l, const allocator_type &__a)
- template<typename _InputIterator >
 map (_InputIterator __first, _InputIterator __last, const allocator_type &__a)
- **map** (const [_Base](#) &__x)
- **map** (const _Compare &__comp, const _Allocator &__a=_Allocator())
- template<typename _InputIterator >
 map (_InputIterator __first, _InputIterator __last, const _Compare &__comp=_Compare(), const _Allocator &__a=_Allocator())
- void [_M_attach](#) (_Safe_iterator_base * __it, bool __constant)
- void [_M_attach_single](#) (_Safe_iterator_base * __it, bool __constant) throw ()
- [_Base](#) & [_M_base](#) () noexcept
- const [_Base](#) & [_M_base](#) () const noexcept
- void [_M_detach](#) (_Safe_iterator_base * __it)
- void [_M_detach_single](#) (_Safe_iterator_base * __it) throw ()
- void [_M_invalidate_all](#) () const
- void [_M_invalidate_if](#) (_Predicate __pred)
- void [_M_swap](#) (_Safe_container &__x) noexcept
- void [_M_transfer_from_if](#) (_Safe_sequence &__from, _Predicate __pred)
- [iterator](#) **begin** () noexcept
- [const_iterator](#) **begin** () const noexcept
- [const_iterator](#) **cbegin** () const noexcept
- [const_iterator](#) **cend** () const noexcept
- void **clear** () noexcept
- [const_reverse_iterator](#) **crbegin** () const noexcept
- [const_reverse_iterator](#) **crend** () const noexcept
- template<typename... _Args>
 [std::pair](#)< [iterator](#), bool > **emplace** (_Args &&...__args)
- template<typename... _Args>
 [iterator](#) **emplace_hint** ([const_iterator](#) __pos, _Args &&...__args)
- [iterator](#) **end** () noexcept
- [const_iterator](#) **end** () const noexcept
- [std::pair](#)< [iterator](#), [iterator](#) > **equal_range** (const key_type &__x)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
 [std::pair](#)< [iterator](#), [iterator](#) > **equal_range** (const _Kt &__x)
- [std::pair](#)< [const_iterator](#), [const_iterator](#) > **equal_range** (const key_type &__x) const
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
 [std::pair](#)< [const_iterator](#), [const_iterator](#) > **equal_range** (const _Kt &__x) const
- [iterator](#) **erase** ([const_iterator](#) __position)
- [iterator](#) **erase** ([iterator](#) __position)
- size_type **erase** (const key_type &__x)
- [iterator](#) **erase** ([const_iterator](#) __first, [const_iterator](#) __last)
- [iterator](#) **find** (const key_type &__x)

- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`iterator find (const _Kt &__x)`
- `const_iterator find (const key_type &__x) const`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`const_iterator find (const _Kt &__x) const`
- `std::pair< iterator, bool > insert (const value_type &__x)`
- `template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>`
`std::pair< iterator, bool > insert (_Pair &&__x)`
- `void insert (std::initializer_list< value_type > __list)`
- `iterator insert (const_iterator __position, const value_type &__x)`
- `template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>`
`iterator insert (const_iterator __position, _Pair &&__x)`
- `template<typename _InputIterator >`
`void insert (_InputIterator __first, _InputIterator __last)`
- `iterator lower_bound (const key_type &__x)`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`iterator lower_bound (const _Kt &__x)`
- `const_iterator lower_bound (const key_type &__x) const`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`const_iterator lower_bound (const _Kt &__x) const`
- `map & operator= (const map &)=default`
- `map & operator= (map &&)=default`
- `map & operator= (initializer_list< value_type > __l)`
- `reverse_iterator rbegin () noexcept`
- `const_reverse_iterator rbegin () const noexcept`
- `reverse_iterator rend () noexcept`
- `const_reverse_iterator rend () const noexcept`
- `void swap (map &__x) noexcept(*conditional *)`
- `iterator upper_bound (const key_type &__x)`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`iterator upper_bound (const _Kt &__x)`
- `const_iterator upper_bound (const key_type &__x) const`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`const_iterator upper_bound (const _Kt &__x) const`

Public Attributes

- `_Safe_iterator_base * _M_const_iterators`
- `_Safe_iterator_base * _M_iterators`
- `unsigned int _M_version`

Protected Member Functions

- `void _M_detach_all ()`
- `void _M_detach_singular ()`
- `__gnu_cxx::__mutex & _M_get_mutex () throw ()`
- `void _M_invalidate_all ()`
- `void _M_revalidate_singular ()`
- `_Safe_container & _M_safe () noexcept`
- `void _M_swap (_Safe_sequence_base &__x) noexcept`

5.439.1 Detailed Description

```
template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Allocator = std::allocator<std::pair<const _Key, _Tp>>>>
class std::__debug::map<_Key, _Tp, _Compare, _Allocator >
```

Class std::map with safety/checking/debug instrumentation.

Definition at line 44 of file debug/map.h.

5.439.2 Member Function Documentation

```
5.439.2.1 void __gnu_debug::Safe_sequence_base::M_attach ( _Safe_iterator_base * __it, bool __constant )
[inherited]
```

Attach an iterator to this sequence.

```
5.439.2.2 void __gnu_debug::Safe_sequence_base::M_attach_single ( _Safe_iterator_base * __it, bool __constant ) throw )
[inherited]
```

Likewise but not thread safe.

```
5.439.2.3 void __gnu_debug::Safe_sequence_base::M_detach ( _Safe_iterator_base * __it ) [inherited]
```

Detach an iterator from this sequence

Referenced by __gnu_debug::Safe_iterator<_Iterator, _Sequence >::Safe_iterator().

```
5.439.2.4 void __gnu_debug::Safe_sequence_base::M_detach_all ( ) [protected], [inherited]
```

Detach all iterators, leaving them singular.

```
5.439.2.5 void __gnu_debug::Safe_sequence_base::M_detach_single ( _Safe_iterator_base * __it ) throw )
[inherited]
```

Likewise but not thread safe.

```
5.439.2.6 void __gnu_debug::Safe_sequence_base::M_detach_singular ( ) [protected], [inherited]
```

Detach all singular iterators.

Postcondition

for all iterators i attached to this sequence, i->_M_version == _M_version.

5.439.2.7 `__gnu_cxx::mutex& __gnu_debug::Safe_sequence_base::M_get_mutex () throw` [protected],
[inherited]

For use in `_Safe_sequence`.

Referenced by `__gnu_debug::Safe_sequence<_Sequence>::M_transfer_from_if()`.

5.439.2.8 `void __gnu_debug::Safe_sequence_base::M_invalidate_all () const` [inline], [inherited]

Invalidates all iterators.

Definition at line 248 of file `safe_base.h`.

References `__gnu_debug::Safe_iterator_base::M_attach()`, `__gnu_debug::Safe_iterator_base::M_attach_single()`, `__gnu_debug::Safe_iterator_base::M_detach()`, and `__gnu_debug::Safe_iterator_base::M_detach_single()`.

5.439.2.9 `void __gnu_debug::Safe_sequence<map<_Key, _Tp, _Compare, _Allocator>>::M_invalidate_if (_Predicate __pred)` [inherited]

Invalidates all iterators `x` that reference this sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

5.439.2.10 `void __gnu_debug::Safe_sequence_base::M_revalidate_singular ()` [protected], [inherited]

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

5.439.2.11 `void __gnu_debug::Safe_sequence_base::M_swap (_Safe_sequence_base & __x)` [protected],
[noexcept], [inherited]

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.439.2.12 `void __gnu_debug::Safe_sequence<map<_Key, _Tp, _Compare, _Allocator>>::M_transfer_from_if (_Safe_sequence<map<_Key, _Tp, _Compare, _Allocator>> & __from, _Predicate __pred)` [inherited]

Transfers all iterators `x` that reference `from` sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

5.439.3 Member Data Documentation

5.439.3.1 `_Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_const_iterators` [inherited]

The list of constant iterators that reference this container.

Definition at line 193 of file `safe_base.h`.

Referenced by `__gnu_debug::Safe_sequence<_Sequence>::M_transfer_from_if()`.

5.439.3.2 __gnu_debug::Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_iterators [inherited]

The list of mutable iterators that reference this container.

Definition at line 190 of file safe_base.h.

Referenced by __gnu_debug::Safe_sequence< _Sequence >::M_transfer_from_if().

5.439.3.3 unsigned int __gnu_debug::Safe_sequence_base::M_version [mutable], [inherited]

The container version number. This number may never be 0.

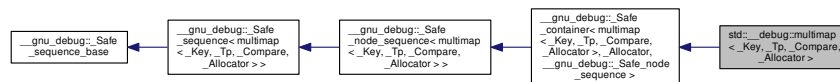
Definition at line 196 of file safe_base.h.

The documentation for this class was generated from the following file:

- [debug/map.h](#)

5.440 std::__debug::multimap< _Key, _Tp, _Compare, _Allocator > Class Template Reference

Inheritance diagram for std::__debug::multimap< _Key, _Tp, _Compare, _Allocator >:



Public Types

- typedef `_Allocator` **allocator_type**
- typedef `__gnu_debug::Safe_iterator< _Base_const_iterator, multimap >` **const_iterator**
- typedef `_Base::const_pointer` **const_pointer**
- typedef `_Base::const_reference` **const_reference**
- typedef `std::reverse_iterator< const_iterator >` **const_reverse_iterator**
- typedef `_Base::difference_type` **difference_type**
- typedef `__gnu_debug::Safe_iterator< _Base_iterator, multimap >` **iterator**
- typedef `_Compare` **key_compare**
- typedef `_Key` **key_type**
- typedef `_Tp` **mapped_type**
- typedef `_Base::pointer` **pointer**
- typedef `_Base::reference` **reference**
- typedef `std::reverse_iterator< iterator >` **reverse_iterator**
- typedef `_Base::size_type` **size_type**
- typedef `std::pair< const _Key, _Tp >` **value_type**

Public Member Functions

- **multimap** (const [multimap](#) &)=default
- **multimap** ([multimap](#) &&)=default
- **multimap** ([initializer_list](#)< [value_type](#) > __l, const [_Compare](#) &__c=[_Compare](#)(), const [allocator_type](#) &__a=[allocator_type](#)())
- **multimap** (const [allocator_type](#) &__a)
- **multimap** (const [multimap](#) &__m, const [allocator_type](#) &__a)
- **multimap** ([multimap](#) &&__m, const [allocator_type](#) &__a)
- **multimap** ([initializer_list](#)< [value_type](#) > __l, const [allocator_type](#) &__a)
- template<typename [_InputIterator](#) >
 multimap ([_InputIterator](#) __first, [_InputIterator](#) __last, const [allocator_type](#) &__a)
- **multimap** (const [_Compare](#) &__comp, const [Allocator](#) &__a=[Allocator](#)())
- template<typename [_InputIterator](#) >
 multimap ([_InputIterator](#) __first, [_InputIterator](#) __last, const [_Compare](#) &__comp=[_Compare](#)(), const [Allocator](#) &__a=[Allocator](#)())
- **multimap** (const [_Base](#) &__x)
- void [_M_attach](#) ([_Safe_iterator_base](#) *__it, bool __constant)
- void [_M_attach_single](#) ([_Safe_iterator_base](#) *__it, bool __constant) throw ()
- [_Base](#) & [_M_base](#) () noexcept
- const [_Base](#) & [_M_base](#) () const noexcept
- void [_M_detach](#) ([_Safe_iterator_base](#) *__it)
- void [_M_detach_single](#) ([_Safe_iterator_base](#) *__it) throw ()
- void [_M_invalidate_all](#) () const
- void [_M_invalidate_if](#) ([_Predicate](#) __pred)
- void [_M_swap](#) ([_Safe_container](#) &__x) noexcept
- void [_M_transfer_from_if](#) ([_Safe_sequence](#) &__from, [_Predicate](#) __pred)
- [iterator](#) [begin](#) () noexcept
- [const_iterator](#) [begin](#) () const noexcept
- [const_iterator](#) [cbegin](#) () const noexcept
- [const_iterator](#) [cend](#) () const noexcept
- void [clear](#) () noexcept
- [const_reverse_iterator](#) [crbegin](#) () const noexcept
- [const_reverse_iterator](#) [crend](#) () const noexcept
- template<typename... [_Args](#)>
 [iterator](#) [emplace](#) ([_Args](#) &&...__args)
- template<typename... [_Args](#)>
 [iterator](#) [emplace_hint](#) ([const_iterator](#) __pos, [_Args](#) &&...__args)
- [iterator](#) [end](#) () noexcept
- [const_iterator](#) [end](#) () const noexcept
- [std::pair](#)< [iterator](#), [iterator](#) > [equal_range](#) (const [key_type](#) &__x)
- template<typename [_Kt](#), typename [_Req](#) = [typename](#) __has_is_transparent< [_Compare](#), [_Kt](#)::[type](#)>
 [std::pair](#)< [iterator](#), [iterator](#) > [equal_range](#) (const [_Kt](#) &__x)
- [std::pair](#)< [const_iterator](#), [const_iterator](#) > [equal_range](#) (const [key_type](#) &__x) const
- template<typename [_Kt](#), typename [_Req](#) = [typename](#) __has_is_transparent< [_Compare](#), [_Kt](#)::[type](#)>
 [std::pair](#)< [const_iterator](#), [const_iterator](#) > [equal_range](#) (const [_Kt](#) &__x) const
- [iterator](#) [erase](#) ([const_iterator](#) __position)
- [iterator](#) [erase](#) ([iterator](#) __position)
- [size_type](#) [erase](#) (const [key_type](#) &__x)
- [iterator](#) [erase](#) ([const_iterator](#) __first, [const_iterator](#) __last)
- [iterator](#) [find](#) (const [key_type](#) &__x)

- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`iterator find (const _Kt &__x)`
- `const_iterator find (const key_type &__x) const`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`const_iterator find (const _Kt &__x) const`
- `iterator insert (const value_type &__x)`
- `template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>`
`iterator insert (_Pair &&__x)`
- `void insert (std::initializer_list< value_type > __list)`
- `iterator insert (const_iterator __position, const value_type &__x)`
- `template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>`
`iterator insert (const_iterator __position, _Pair &&__x)`
- `template<typename _InputIterator >`
`void insert (_InputIterator __first, _InputIterator __last)`
- `iterator lower_bound (const key_type &__x)`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`iterator lower_bound (const _Kt &__x)`
- `const_iterator lower_bound (const key_type &__x) const`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`const_iterator lower_bound (const _Kt &__x) const`
- `multimap & operator= (const multimap &)=default`
- `multimap & operator= (multimap &&)=default`
- `multimap & operator= (initializer_list< value_type > __l)`
- `reverse_iterator rbegin () noexcept`
- `const_reverse_iterator rbegin () const noexcept`
- `reverse_iterator rend () noexcept`
- `const_reverse_iterator rend () const noexcept`
- `void swap (multimap &__x) noexcept (/*conditional */)`
- `iterator upper_bound (const key_type &__x)`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`iterator upper_bound (const _Kt &__x)`
- `const_iterator upper_bound (const key_type &__x) const`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`const_iterator upper_bound (const _Kt &__x) const`

Public Attributes

- `_Safe_iterator_base * _M_const_iterators`
- `_Safe_iterator_base * _M_iterators`
- `unsigned int _M_version`

Protected Member Functions

- `void _M_detach_all ()`
- `void _M_detach_singular ()`
- `__gnu_cxx::__mutex & _M_get_mutex () throw ()`
- `void _M_invalidate_all ()`
- `void _M_revalidate_singular ()`
- `_Safe_container & _M_safe () noexcept`
- `void _M_swap (_Safe_sequence_base &__x) noexcept`

5.440.1 Detailed Description

```
template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Allocator = std::allocator<std::pair<const _Key, _Tp>>>>
class std::__debug::multimap<_Key, _Tp, _Compare, _Allocator>
```

Class std::multimap with safety/checking/debug instrumentation.

Definition at line 44 of file debug/multimap.h.

5.440.2 Member Function Documentation

```
5.440.2.1 void __gnu_debug::Safe_sequence_base::M_attach ( _Safe_iterator_base * __it, bool __constant )
[inherited]
```

Attach an iterator to this sequence.

```
5.440.2.2 void __gnu_debug::Safe_sequence_base::M_attach_single ( _Safe_iterator_base * __it, bool __constant ) throw )
[inherited]
```

Likewise but not thread safe.

```
5.440.2.3 void __gnu_debug::Safe_sequence_base::M_detach ( _Safe_iterator_base * __it ) [inherited]
```

Detach an iterator from this sequence

Referenced by __gnu_debug::Safe_iterator<_Iterator, _Sequence>::Safe_iterator().

```
5.440.2.4 void __gnu_debug::Safe_sequence_base::M_detach_all ( ) [protected], [inherited]
```

Detach all iterators, leaving them singular.

```
5.440.2.5 void __gnu_debug::Safe_sequence_base::M_detach_single ( _Safe_iterator_base * __it ) throw )
[inherited]
```

Likewise but not thread safe.

```
5.440.2.6 void __gnu_debug::Safe_sequence_base::M_detach_singular ( ) [protected], [inherited]
```

Detach all singular iterators.

Postcondition

for all iterators *i* attached to this sequence, *i*->_M_version == _M_version.

5.440.2.7 `__gnu_cxx::mutex& __gnu_debug::Safe_sequence_base::M_get_mutex () throw` `[protected]`,
`[inherited]`

For use in `_Safe_sequence`.

Referenced by `__gnu_debug::Safe_sequence<_Sequence>::M_transfer_from_if()`.

5.440.2.8 `void __gnu_debug::Safe_sequence_base::M_invalidate_all () const` `[inline]`, `[inherited]`

Invalidates all iterators.

Definition at line 248 of file `safe_base.h`.

References `__gnu_debug::Safe_iterator_base::M_attach()`, `__gnu_debug::Safe_iterator_base::M_attach_single()`, `__gnu_debug::Safe_iterator_base::M_detach()`, and `__gnu_debug::Safe_iterator_base::M_detach_single()`.

5.440.2.9 `void __gnu_debug::Safe_sequence<multimap<_Key,_Tp,_Compare,_Allocator>>::M_invalidate_if (`
`_Predicate __pred)` `[inherited]`

Invalidates all iterators `x` that reference this sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

5.440.2.10 `void __gnu_debug::Safe_sequence_base::M_revalidate_singular ()` `[protected]`, `[inherited]`

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

5.440.2.11 `void __gnu_debug::Safe_sequence_base::M_swap (_Safe_sequence_base & __x)` `[protected]`,
`[noexcept]`, `[inherited]`

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.440.2.12 `void __gnu_debug::Safe_sequence<multimap<_Key,_Tp,_Compare,_Allocator>>::M_transfer_from_if`
`(_Safe_sequence<multimap<_Key,_Tp,_Compare,_Allocator>> & __from, _Predicate __pred)`
`[inherited]`

Transfers all iterators `x` that reference `from` sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

5.440.3 Member Data Documentation

5.440.3.1 `_Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_const_iterators` `[inherited]`

The list of constant iterators that reference this container.

Definition at line 193 of file `safe_base.h`.

Referenced by `__gnu_debug::Safe_sequence<_Sequence>::M_transfer_from_if()`.

5.440.3.2 `_Safe_iterator_base* __gnu_debug::_Safe_sequence_base::_M_iterators` [inherited]

The list of mutable iterators that reference this container.

Definition at line 190 of file `safe_base.h`.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`.

5.440.3.3 `unsigned int __gnu_debug::_Safe_sequence_base::_M_version` [mutable],[inherited]

The container version number. This number may never be 0.

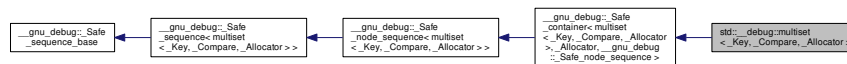
Definition at line 196 of file `safe_base.h`.

The documentation for this class was generated from the following file:

- [debug/multimap.h](#)

5.441 `std::__debug::multiset<_Key, _Compare, _Allocator>` Class Template Reference

Inheritance diagram for `std::__debug::multiset<_Key, _Compare, _Allocator>`:



Public Types

- typedef `_Allocator` **allocator_type**
- typedef `__gnu_debug::_Safe_iterator<_Base_const_iterator, multiset>` **const_iterator**
- typedef `_Base::const_pointer` **const_pointer**
- typedef `_Base::const_reference` **const_reference**
- typedef `std::reverse_iterator<const_iterator>` **const_reverse_iterator**
- typedef `_Base::difference_type` **difference_type**
- typedef `__gnu_debug::_Safe_iterator<_Base_iterator, multiset>` **iterator**
- typedef `_Compare` **key_compare**
- typedef `_Key` **key_type**
- typedef `_Base::pointer` **pointer**
- typedef `_Base::reference` **reference**
- typedef `std::reverse_iterator<iterator>` **reverse_iterator**
- typedef `_Base::size_type` **size_type**
- typedef `_Compare` **value_compare**
- typedef `_Key` **value_type**

Public Member Functions

- **multiset** (const [multiset](#) &)=default
- **multiset** ([multiset](#) &&)=default
- **multiset** ([initializer_list](#)< value_type > __l, const __Compare &__comp=_Compare(), const allocator_type &__a=allocator_type())
- **multiset** (const allocator_type &__a)
- **multiset** (const [multiset](#) &__m, const allocator_type &__a)
- **multiset** ([multiset](#) &&__m, const allocator_type &__a)
- **multiset** ([initializer_list](#)< value_type > __l, const allocator_type &__a)
- template<typename __InputIterator >
multiset (__InputIterator __first, __InputIterator __last, const allocator_type &__a)
- **multiset** (const __Compare &__comp, const __Allocator &__a=_Allocator())
- template<typename __InputIterator >
multiset (__InputIterator __first, __InputIterator __last, const __Compare &__comp=_Compare(), const __Allocator &__a=_Allocator())
- **multiset** (const [_Base](#) &__x)
- void [_M_attach](#) (__Safe_iterator_base *__it, bool __constant)
- void [_M_attach_single](#) (__Safe_iterator_base *__it, bool __constant) throw ()
- [_Base](#) & [_M_base](#) () noexcept
- const [_Base](#) & [_M_base](#) () const noexcept
- void [_M_detach](#) (__Safe_iterator_base *__it)
- void [_M_detach_single](#) (__Safe_iterator_base *__it) throw ()
- void [_M_invalidate_all](#) () const
- void [_M_invalidate_if](#) (__Predicate __pred)
- void [_M_swap](#) (__Safe_container &__x) noexcept
- void [_M_transfer_from_if](#) (__Safe_sequence &__from, __Predicate __pred)
- [iterator](#) **begin** () noexcept
- const [iterator](#) **begin** () const noexcept
- const [iterator](#) **cbegin** () const noexcept
- const [iterator](#) **cend** () const noexcept
- void **clear** () noexcept
- const [reverse_iterator](#) **crbegin** () const noexcept
- const [reverse_iterator](#) **crend** () const noexcept
- template<typename... __Args>
[iterator](#) **emplace** (__Args &&...__args)
- template<typename... __Args>
[iterator](#) **emplace_hint** (const [iterator](#) __pos, __Args &&...__args)
- [iterator](#) **end** () noexcept
- const [iterator](#) **end** () const noexcept
- std::pair< [iterator](#), [iterator](#) > **equal_range** (const key_type &__x)
- std::pair< const [iterator](#), const [iterator](#) > **equal_range** (const key_type &__x) const
- template<typename __Kt, typename __Req = typename __has_is_transparent< __Compare, __Kt>::type>
std::pair< [iterator](#), [iterator](#) > **equal_range** (const __Kt &__x)
- template<typename __Kt, typename __Req = typename __has_is_transparent< __Compare, __Kt>::type>
std::pair< const [iterator](#), const [iterator](#) > **equal_range** (const __Kt &__x) const
- [iterator](#) **erase** (const [iterator](#) __position)
- size_type **erase** (const key_type &__x)
- [iterator](#) **erase** (const [iterator](#) __first, const [iterator](#) __last)
- [iterator](#) **find** (const key_type &__x)
- const [iterator](#) **find** (const key_type &__x) const

- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`iterator find (const _Kt &__x)`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`const_iterator find (const _Kt &__x) const`
- `iterator insert (const value_type &__x)`
- `iterator insert (value_type &&__x)`
- `iterator insert (const_iterator __position, const value_type &__x)`
- `iterator insert (const_iterator __position, value_type &&__x)`
- `template<typename _InputIterator >`
`void insert (_InputIterator __first, _InputIterator __last)`
- `void insert (initializer_list< value_type > __l)`
- `iterator lower_bound (const key_type &__x)`
- `const_iterator lower_bound (const key_type &__x) const`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`iterator lower_bound (const _Kt &__x)`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`const_iterator lower_bound (const _Kt &__x) const`
- `multiset & operator= (const multiset &)=default`
- `multiset & operator= (multiset &&)=default`
- `multiset & operator= (initializer_list< value_type > __l)`
- `reverse_iterator rbegin () noexcept`
- `const_reverse_iterator rbegin () const noexcept`
- `reverse_iterator rend () noexcept`
- `const_reverse_iterator rend () const noexcept`
- `void swap (multiset &__x) noexcept(*conditional *)`
- `iterator upper_bound (const key_type &__x)`
- `const_iterator upper_bound (const key_type &__x) const`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`iterator upper_bound (const _Kt &__x)`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`const_iterator upper_bound (const _Kt &__x) const`

Public Attributes

- `_Safe_iterator_base * _M_const_iterators`
- `_Safe_iterator_base * _M_iterators`
- `unsigned int _M_version`

Protected Member Functions

- `void _M_detach_all ()`
- `void _M_detach_singular ()`
- `__gnu_cxx::__mutex & _M_get_mutex () throw ()`
- `void _M_invalidate_all ()`
- `void _M_revalidate_singular ()`
- `_Safe_container & _M_safe () noexcept`
- `void _M_swap (_Safe_sequence_base &__x) noexcept`

5.441.1 Detailed Description

```
template<typename _Key, typename _Compare = std::less<_Key>, typename _Allocator = std::allocator<_Key>>
class std::__debug::multiset< _Key, _Compare, _Allocator >
```

Class std::multiset with safety/checking/debug instrumentation.

Definition at line 44 of file debug/multiset.h.

5.441.2 Member Function Documentation

```
5.441.2.1 void __gnu_debug::Safe_sequence_base::M_attach ( _Safe_iterator_base * __it, bool __constant )
[inherited]
```

Attach an iterator to this sequence.

```
5.441.2.2 void __gnu_debug::Safe_sequence_base::M_attach_single ( _Safe_iterator_base * __it, bool __constant ) throw )
[inherited]
```

Likewise but not thread safe.

```
5.441.2.3 void __gnu_debug::Safe_sequence_base::M_detach ( _Safe_iterator_base * __it ) [inherited]
```

Detach an iterator from this sequence

Referenced by __gnu_debug::Safe_iterator< _Iterator, _Sequence >::_Safe_iterator().

```
5.441.2.4 void __gnu_debug::Safe_sequence_base::M_detach_all ( ) [protected], [inherited]
```

Detach all iterators, leaving them singular.

```
5.441.2.5 void __gnu_debug::Safe_sequence_base::M_detach_single ( _Safe_iterator_base * __it ) throw )
[inherited]
```

Likewise but not thread safe.

```
5.441.2.6 void __gnu_debug::Safe_sequence_base::M_detach_singular ( ) [protected], [inherited]
```

Detach all singular iterators.

Postcondition

for all iterators *i* attached to this sequence, *i*->_M_version == _M_version.

5.441.2.7 `__gnu_cxx::mutex& __gnu_debug::Safe_sequence_base::M_get_mutex () throw` [protected],
[inherited]

For use in `_Safe_sequence`.

Referenced by `__gnu_debug::Safe_sequence<_Sequence>::M_transfer_from_if()`.

5.441.2.8 `void __gnu_debug::Safe_sequence_base::M_invalidate_all () const` [inline], [inherited]

Invalidates all iterators.

Definition at line 248 of file `safe_base.h`.

References `__gnu_debug::Safe_iterator_base::M_attach()`, `__gnu_debug::Safe_iterator_base::M_attach_single()`, `__gnu_debug::Safe_iterator_base::M_detach()`, and `__gnu_debug::Safe_iterator_base::M_detach_single()`.

5.441.2.9 `void __gnu_debug::Safe_sequence<multiset<_Key,_Compare,_Allocator>>::M_invalidate_if (_Predicate __pred)` [inherited]

Invalidates all iterators `x` that reference this sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

5.441.2.10 `void __gnu_debug::Safe_sequence_base::M_revalidate_singular ()` [protected], [inherited]

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

5.441.2.11 `void __gnu_debug::Safe_sequence_base::M_swap (_Safe_sequence_base & __x)` [protected],
[noexcept], [inherited]

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.441.2.12 `void __gnu_debug::Safe_sequence<multiset<_Key,_Compare,_Allocator>>::M_transfer_from_if (_Safe_sequence<multiset<_Key,_Compare,_Allocator>> & __from, _Predicate __pred)` [inherited]

Transfers all iterators `x` that reference `from` sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

5.441.3 Member Data Documentation

5.441.3.1 `_Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_const_iterators` [inherited]

The list of constant iterators that reference this container.

Definition at line 193 of file `safe_base.h`.

Referenced by `__gnu_debug::Safe_sequence<_Sequence>::M_transfer_from_if()`.

5.441.3.2 `_Safe_iterator_base* __gnu_debug::_Safe_sequence_base::M_iterators` [inherited]

The list of mutable iterators that reference this container.

Definition at line 190 of file safe_base.h.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`.

5.441.3.3 unsigned int __gnu_debug::_Safe_sequence_base::_M_version [mutable], [inherited]

The container version number. This number may never be 0.

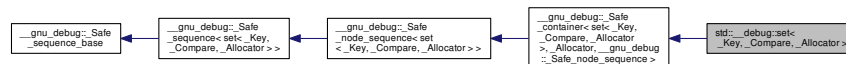
Definition at line 196 of file safe_base.h.

The documentation for this class was generated from the following file:

- `debug/multiset.h`

5.442 `std::__debug::set<_Key, _Compare, _Allocator>` Class Template Reference

Inheritance diagram for `std::__debug::set<_Key, _Compare, _Allocator>`:



Public Types

- typedef `_Allocator` **allocator_type**
- typedef `__gnu_debug::_Safe_iterator< _Base_const_iterator, set >` **const_iterator**
- typedef `_Base::const_pointer` **const_pointer**
- typedef `_Base::const_reference` **const_reference**
- typedef `std::reverse_iterator< const_iterator >` **const_reverse_iterator**
- typedef `_Base::difference_type` **difference_type**
- typedef `__gnu_debug::_Safe_iterator< _Base_iterator, set >` **iterator**
- typedef `_Compare` **key_compare**
- typedef `_Key` **key_type**
- typedef `_Base::pointer` **pointer**
- typedef `_Base::reference` **reference**
- typedef `std::reverse_iterator< iterator >` **reverse_iterator**
- typedef `_Base::size_type` **size_type**
- typedef `_Compare` **value_compare**
- typedef `_Key` **value_type**

Public Member Functions

- **set** (const [set](#) &)=default
- **set** ([set](#) &&)=default
- **set** ([initializer_list](#)< value_type > __l, const _Compare &__comp=_Compare(), const allocator_type &__a=allocator_type())
- **set** (const allocator_type &__a)
- **set** (const [set](#) &__x, const allocator_type &__a)
- **set** ([set](#) &&__x, const allocator_type &__a)
- **set** ([initializer_list](#)< value_type > __l, const allocator_type &__a)
- template<typename _InputIterator >
 set (_InputIterator __first, _InputIterator __last, const allocator_type &__a)
- **set** (const _Compare &__comp, const _Allocator &__a=_Allocator())
- template<typename _InputIterator >
 set (_InputIterator __first, _InputIterator __last, const _Compare &__comp=_Compare(), const _Allocator &__a=_Allocator())
- **set** (const [_Base](#) &__x)
- void [_M_attach](#) (_Safe_iterator_base *__it, bool __constant)
- void [_M_attach_single](#) (_Safe_iterator_base *__it, bool __constant) throw ()
- [_Base](#) & [_M_base](#) () noexcept
- const [_Base](#) & [_M_base](#) () const noexcept
- void [_M_detach](#) (_Safe_iterator_base *__it)
- void [_M_detach_single](#) (_Safe_iterator_base *__it) throw ()
- void [_M_invalidate_all](#) () const
- void [_M_invalidate_if](#) (_Predicate __pred)
- void [_M_swap](#) (_Safe_container &__x) noexcept
- void [_M_transfer_from_if](#) (_Safe_sequence &__from, _Predicate __pred)
- [iterator begin](#) () noexcept
- [const_iterator begin](#) () const noexcept
- [const_iterator cbegin](#) () const noexcept
- [const_iterator cend](#) () const noexcept
- void [clear](#) () noexcept
- [const_reverse_iterator crbegin](#) () const noexcept
- [const_reverse_iterator crend](#) () const noexcept
- template<typename... _Args>
 std::pair< [iterator](#), bool > **emplace** (_Args &&...__args)
- template<typename... _Args>
 [iterator](#) **emplace_hint** ([const_iterator](#) __pos, _Args &&...__args)
- [iterator end](#) () noexcept
- [const_iterator end](#) () const noexcept
- std::pair< [iterator](#), [iterator](#) > **equal_range** (const key_type &__x)
- std::pair< [const_iterator](#), [const_iterator](#) > **equal_range** (const key_type &__x) const
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
 std::pair< [iterator](#), [iterator](#) > **equal_range** (const _Kt &__x)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
 std::pair< [const_iterator](#), [const_iterator](#) > **equal_range** (const _Kt &__x) const
- [iterator erase](#) ([const_iterator](#) __position)
- size_type [erase](#) (const key_type &__x)
- [iterator erase](#) ([const_iterator](#) __first, [const_iterator](#) __last)
- [iterator find](#) (const key_type &__x)
- [const_iterator find](#) (const key_type &__x) const

- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`iterator find (const _Kt &__x)`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`const_iterator find (const _Kt &__x) const`
- `std::pair< iterator, bool > insert (const value_type &__x)`
- `std::pair< iterator, bool > insert (value_type &&__x)`
- `iterator insert (const_iterator __position, const value_type &__x)`
- `iterator insert (const_iterator __position, value_type &&__x)`
- `template<typename _InputIterator >`
`void insert (_InputIterator __first, _InputIterator __last)`
- `void insert (initializer_list< value_type > __l)`
- `iterator lower_bound (const key_type &__x)`
- `const_iterator lower_bound (const key_type &__x) const`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`iterator lower_bound (const _Kt &__x)`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`const_iterator lower_bound (const _Kt &__x) const`
- `set & operator= (const set &)=default`
- `set & operator= (set &&)=default`
- `set & operator= (initializer_list< value_type > __l)`
- `reverse_iterator rbegin () noexcept`
- `const_reverse_iterator rbegin () const noexcept`
- `reverse_iterator rend () noexcept`
- `const_reverse_iterator rend () const noexcept`
- `void swap (set &__x) noexcept(*conditional *)`
- `iterator upper_bound (const key_type &__x)`
- `const_iterator upper_bound (const key_type &__x) const`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`iterator upper_bound (const _Kt &__x)`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`const_iterator upper_bound (const _Kt &__x) const`

Public Attributes

- `_Safe_iterator_base * _M_const_iterators`
- `_Safe_iterator_base * _M_iterators`
- `unsigned int _M_version`

Protected Member Functions

- `void _M_detach_all ()`
- `void _M_detach_singular ()`
- `__gnu_cxx::__mutex & _M_get_mutex () throw ()`
- `void _M_invalidate_all ()`
- `void _M_revalidate_singular ()`
- `_Safe_container & _M_safe () noexcept`
- `void _M_swap (_Safe_sequence_base &__x) noexcept`

5.442.1 Detailed Description

```
template<typename _Key, typename _Compare = std::less<_Key>, typename _Allocator = std::allocator<_Key>>
class std::__debug::set< _Key, _Compare, _Allocator >
```

Class std::set with safety/checking/debug instrumentation.

Definition at line 44 of file debug/set.h.

5.442.2 Member Function Documentation

5.442.2.1 void __gnu_debug::Safe_sequence_base::M_attach (_Safe_iterator_base * __it, bool __constant)
[inherited]

Attach an iterator to this sequence.

5.442.2.2 void __gnu_debug::Safe_sequence_base::M_attach_single (_Safe_iterator_base * __it, bool __constant) throw)
[inherited]

Likewise but not thread safe.

5.442.2.3 void __gnu_debug::Safe_sequence_base::M_detach (_Safe_iterator_base * __it) [inherited]

Detach an iterator from this sequence

Referenced by __gnu_debug::Safe_iterator< _Iterator, _Sequence >::Safe_iterator().

5.442.2.4 void __gnu_debug::Safe_sequence_base::M_detach_all () [protected], [inherited]

Detach all iterators, leaving them singular.

5.442.2.5 void __gnu_debug::Safe_sequence_base::M_detach_single (_Safe_iterator_base * __it) throw)
[inherited]

Likewise but not thread safe.

5.442.2.6 void __gnu_debug::Safe_sequence_base::M_detach_singular () [protected], [inherited]

Detach all singular iterators.

Postcondition

for all iterators i attached to this sequence, i->_M_version == _M_version.

5.442.2.7 `__gnu_cxx::mutex& __gnu_debug::Safe_sequence_base::M_get_mutex () throw` [protected],
[inherited]

For use in `_Safe_sequence`.

Referenced by `__gnu_debug::Safe_sequence<_Sequence >::M_transfer_from_if()`.

5.442.2.8 `void __gnu_debug::Safe_sequence_base::M_invalidate_all () const` [inline], [inherited]

Invalidates all iterators.

Definition at line 248 of file `safe_base.h`.

References `__gnu_debug::Safe_iterator_base::M_attach()`, `__gnu_debug::Safe_iterator_base::M_attach_single()`, `__gnu_debug::Safe_iterator_base::M_detach()`, and `__gnu_debug::Safe_iterator_base::M_detach_single()`.

5.442.2.9 `void __gnu_debug::Safe_sequence<set<_Key, _Compare, _Allocator >>::M_invalidate_if (_Predicate __pred)` [inherited]

Invalidates all iterators `x` that reference this sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

5.442.2.10 `void __gnu_debug::Safe_sequence_base::M_revalidate_singular ()` [protected], [inherited]

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

5.442.2.11 `void __gnu_debug::Safe_sequence_base::M_swap (_Safe_sequence_base & __x)` [protected],
[noexcept], [inherited]

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.442.2.12 `void __gnu_debug::Safe_sequence<set<_Key, _Compare, _Allocator >>::M_transfer_from_if (_Safe_sequence<set<_Key, _Compare, _Allocator >> & __from, _Predicate __pred)` [inherited]

Transfers all iterators `x` that reference `from` sequence, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

5.442.3 Member Data Documentation

5.442.3.1 `_Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_const_iterators` [inherited]

The list of constant iterators that reference this container.

Definition at line 193 of file `safe_base.h`.

Referenced by `__gnu_debug::Safe_sequence<_Sequence >::M_transfer_from_if()`.

5.442.3.2 `_Safe_iterator_base* __gnu_debug::_Safe_sequence_base::_M_iterators` [inherited]

The list of mutable iterators that reference this container.

Definition at line 190 of file `safe_base.h`.

Referenced by `__gnu_debug::_Safe_sequence<_Sequence>::_M_transfer_from_if()`.

5.442.3.3 `unsigned int __gnu_debug::_Safe_sequence_base::_M_version` [mutable],[inherited]

The container version number. This number may never be 0.

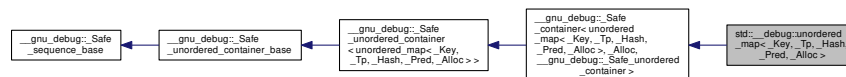
Definition at line 196 of file `safe_base.h`.

The documentation for this class was generated from the following file:

- [debug/set.h](#)

5.443 `std::__debug::unordered_map<_Key,_Tp,_Hash,_Pred,_Alloc>` Class Template Reference

Inheritance diagram for `std::__debug::unordered_map<_Key,_Tp,_Hash,_Pred,_Alloc>`:



Public Types

- typedef `_Base::allocator_type` **allocator_type**
- typedef `__gnu_debug::_Safe_iterator<_Base_const_iterator, unordered_map>` **const_iterator**
- typedef `__gnu_debug::_Safe_local_iterator<_Base_const_local_iterator, unordered_map>` **const_local_iterator**
- typedef `_Base::hasher` **hasher**
- typedef `__gnu_debug::_Safe_iterator<_Base_iterator, unordered_map>` **iterator**
- typedef `_Base::key_equal` **key_equal**
- typedef `_Base::key_type` **key_type**
- typedef `__gnu_debug::_Safe_local_iterator<_Base_local_iterator, unordered_map>` **local_iterator**
- typedef `_Base::size_type` **size_type**
- typedef `_Base::value_type` **value_type**

Public Member Functions

- **unordered_map** (size_type __n, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- template<typename _InputIterator >
unordered_map (_InputIterator __first, _InputIterator __last, size_type __n=0, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- **unordered_map** (const [unordered_map](#) &)=default
- **unordered_map** (const [_Base](#) &__x)
- **unordered_map** ([unordered_map](#) &&)=default
- **unordered_map** (const allocator_type &__a)
- **unordered_map** (const [unordered_map](#) &__umap, const allocator_type &__a)
- **unordered_map** ([unordered_map](#) &&__umap, const allocator_type &__a)
- **unordered_map** ([initializer_list](#)< value_type > __l, size_type __n=0, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- **unordered_map** (size_type __n, const allocator_type &__a)
- **unordered_map** (size_type __n, const hasher &__hf, const allocator_type &__a)
- template<typename _InputIterator >
unordered_map (_InputIterator __first, _InputIterator __last, size_type __n, const allocator_type &__a)
- template<typename _InputIterator >
unordered_map (_InputIterator __first, _InputIterator __last, size_type __n, const hasher &__hf, const allocator_type &__a)
- **unordered_map** ([initializer_list](#)< value_type > __l, size_type __n, const allocator_type &__a)
- **unordered_map** ([initializer_list](#)< value_type > __l, size_type __n, const hasher &__hf, const allocator_type &__a)
- void [_M_attach](#) (_Safe_iterator_base *__it, bool __constant)
- void [_M_attach_local](#) (_Safe_iterator_base *__it, bool __constant)
- void [_M_attach_local_single](#) (_Safe_iterator_base *__it, bool __constant) throw ()
- void [_M_attach_single](#) (_Safe_iterator_base *__it, bool __constant) throw ()
- [_Base](#) & [_M_base](#) () noexcept
- const [_Base](#) & [_M_base](#) () const noexcept
- void [_M_detach](#) (_Safe_iterator_base *__it)
- void [_M_detach_local](#) (_Safe_iterator_base *__it)
- void [_M_detach_local_single](#) (_Safe_iterator_base *__it) throw ()
- void [_M_detach_single](#) (_Safe_iterator_base *__it) throw ()
- void [_M_invalidate_all](#) () const
- void [_M_swap](#) (_Safe_container &__x) noexcept
- [iterator](#) [begin](#) () noexcept
- const [iterator](#) [begin](#) () const noexcept
- [local_iterator](#) [begin](#) (size_type __b)
- const [local_iterator](#) [begin](#) (size_type __b) const
- size_type [bucket_size](#) (size_type __b) const
- const [iterator](#) [cbegin](#) () const noexcept
- const [local_iterator](#) [cbegin](#) (size_type __b) const
- const [iterator](#) [cend](#) () const noexcept
- const [local_iterator](#) [cend](#) (size_type __b) const
- void [clear](#) () noexcept
- template<typename... _Args>
[std::pair](#)< [iterator](#), bool > [emplace](#) (_Args &&...__args)
- template<typename... _Args>
[iterator](#) [emplace_hint](#) (const [iterator](#) __hint, _Args &&...__args)

- `iterator end ()` noexcept
- `const_iterator end ()` const noexcept
- `local_iterator end (size_type __b)`
- `const_local_iterator end (size_type __b)` const
- `std::pair< iterator, iterator > equal_range (const key_type &__key)`
- `std::pair< const_iterator, const_iterator > equal_range (const key_type &__key)` const
- `size_type erase (const key_type &__key)`
- `iterator erase (const_iterator __it)`
- `iterator erase (iterator __it)`
- `iterator erase (const_iterator __first, const_iterator __last)`
- `iterator find (const key_type &__key)`
- `const_iterator find (const key_type &__key)` const
- `std::pair< iterator, bool > insert (const value_type &__obj)`
- `iterator insert (const_iterator __hint, const value_type &__obj)`
- `template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type> std::pair< iterator, bool > insert (_Pair &&__obj)`
- `template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type> iterator insert (const_iterator __hint, _Pair &&__obj)`
- `void insert (std::initializer_list< value_type > __l)`
- `template<typename _InputIterator > void insert (_InputIterator __first, _InputIterator __last)`
- `float max_load_factor ()` const noexcept
- `void max_load_factor (float __f)`
- `unordered_map & operator= (const unordered_map &)=default`
- `unordered_map & operator= (unordered_map &&)=default`
- `unordered_map & operator= (initializer_list< value_type > __l)`
- `void swap (unordered_map &__x)` noexcept(noexcept(declval< _Base & >().swap(__x)))

Public Attributes

- `_Safe_iterator_base * _M_const_iterators`
- `_Safe_iterator_base * _M_const_local_iterators`
- `_Safe_iterator_base * _M_iterators`
- `_Safe_iterator_base * _M_local_iterators`
- `unsigned int _M_version`

Protected Member Functions

- `void _M_detach_all ()`
- `void _M_detach_singular ()`
- `__gnu_cxx::__mutex & _M_get_mutex ()` throw ()
- `void _M_invalidate_all ()`
- `void _M_invalidate_if (_Predicate __pred)`
- `void _M_invalidate_local_if (_Predicate __pred)`
- `void _M_invalidate_locals ()`
- `void _M_revalidate_singular ()`
- `_Safe_container & _M_safe ()` noexcept
- `void _M_swap (_Safe_unordered_container_base &__x)` noexcept
- `void _M_swap (_Safe_sequence_base &__x)` noexcept

5.443.1 Detailed Description

```
template<typename _Key, typename _Tp, typename _Hash = std::hash<_Key>, typename _Pred = std::equal_to<_Key>, typename
_Alloc = std::allocator<std::pair<const _Key, _Tp> >>
class std::__debug::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >
```

Class std::unordered_map with safety/checking/debug instrumentation.

Definition at line 51 of file debug/unordered_map.

5.443.2 Member Function Documentation

5.443.2.1 void __gnu_debug::Safe_sequence_base::M_attach (_Safe_iterator_base * __it, bool __constant)
[inherited]

Attach an iterator to this sequence.

5.443.2.2 void __gnu_debug::Safe_unordered_container_base::M_attach_local (_Safe_iterator_base * __it, bool __constant) [inherited]

Attach an iterator to this container.

5.443.2.3 void __gnu_debug::Safe_unordered_container_base::M_attach_local_single (_Safe_iterator_base * __it, bool __constant) throw) [inherited]

Likewise but not thread safe.

5.443.2.4 void __gnu_debug::Safe_sequence_base::M_attach_single (_Safe_iterator_base * __it, bool __constant) throw) [inherited]

Likewise but not thread safe.

5.443.2.5 void __gnu_debug::Safe_sequence_base::M_detach (_Safe_iterator_base * __it) [inherited]

Detach an iterator from this sequence

Referenced by __gnu_debug::Safe_iterator<_Iterator, _Sequence >::Safe_iterator().

5.443.2.6 void __gnu_debug::Safe_unordered_container_base::M_detach_all () [protected],[inherited]

Detach all iterators, leaving them singular.

5.443.2.7 void __gnu_debug::Safe_unordered_container_base::M_detach_local (_Safe_iterator_base * __it) [inherited]

Detach an iterator from this container

5.443.2.8 void __gnu_debug::Safe_unordered_container_base::M_detach_local_single (_Safe_iterator_base * __it) throw (
 [inherited]

Likewise but not thread safe.

5.443.2.9 void __gnu_debug::Safe_sequence_base::M_detach_single (_Safe_iterator_base * __it) throw (
 [inherited]

Likewise but not thread safe.

5.443.2.10 void __gnu_debug::Safe_sequence_base::M_detach_singular () [protected],[inherited]

Detach all singular iterators.

Postcondition

for all iterators i attached to this sequence, i->_M_version == _M_version.

5.443.2.11 __gnu_cxx::mutex& __gnu_debug::Safe_sequence_base::M_get_mutex () throw ([protected],
 [inherited]

For use in _Safe_sequence.

Referenced by __gnu_debug::Safe_sequence< _Sequence >::M_transfer_from_if().

5.443.2.12 void __gnu_debug::Safe_sequence_base::M_invalidate_all () const [inline],[inherited]

Invalidates all iterators.

Definition at line 248 of file safe_base.h.

References __gnu_debug::Safe_iterator_base::M_attach(), __gnu_debug::Safe_iterator_base::M_attach_single(),
 __gnu_debug::Safe_iterator_base::M_detach(), and __gnu_debug::Safe_iterator_base::M_detach_single().

5.443.2.13 void __gnu_debug::Safe_unordered_container< unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >
 >::M_invalidate_if (_Predicate __pred) [protected],[inherited]

Invalidates all iterators x that reference this container, are not singular, and for which __pred(x) returns true.
 __pred will be invoked with the normal iterators nested in the safe ones.

5.443.2.14 void __gnu_debug::Safe_unordered_container< unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >
 >::M_invalidate_local_if (_Predicate __pred) [protected],[inherited]

Invalidates all local iterators x that reference this container, are not singular, and for which __pred(x) returns true.
 __pred will be invoked with the normal ilocal iterators nested in the safe ones.

5.443.2.15 void __gnu_debug::Safe_sequence_base::M_revalidate_singular () [protected], [inherited]

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

5.443.2.16 void __gnu_debug::Safe_unordered_container_base::M_swap (_Safe_unordered_container_base & __x) [protected], [noexcept], [inherited]

Swap this container with the given container. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.443.2.17 void __gnu_debug::Safe_sequence_base::M_swap (_Safe_sequence_base & __x) [protected], [noexcept], [inherited]

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.443.3 Member Data Documentation

5.443.3.1 _Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_const_iterators [inherited]

The list of constant iterators that reference this container.

Definition at line 193 of file safe_base.h.

Referenced by __gnu_debug::Safe_sequence<_Sequence >::M_transfer_from_if().

5.443.3.2 _Safe_iterator_base* __gnu_debug::Safe_unordered_container_base::M_const_local_iterators [inherited]

The list of constant local iterators that reference this container.

Definition at line 125 of file safe_unordered_base.h.

5.443.3.3 _Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_iterators [inherited]

The list of mutable iterators that reference this container.

Definition at line 190 of file safe_base.h.

Referenced by __gnu_debug::Safe_sequence<_Sequence >::M_transfer_from_if().

5.443.3.4 _Safe_iterator_base* __gnu_debug::Safe_unordered_container_base::M_local_iterators [inherited]

The list of mutable local iterators that reference this container.

Definition at line 122 of file safe_unordered_base.h.

5.443.3.5 `unsigned int __gnu_debug::Safe_sequence_base::M_version` [mutable],[inherited]

The container version number. This number may never be 0.

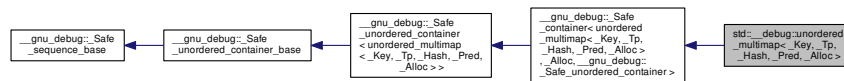
Definition at line 196 of file `safe_base.h`.

The documentation for this class was generated from the following file:

- [debug/unordered_map](#)

5.444 `std::__debug::unordered_multimap<_Key,_Tp,_Hash,_Pred,_Alloc>` Class Template Reference

Inheritance diagram for `std::__debug::unordered_multimap<_Key,_Tp,_Hash,_Pred,_Alloc>`:



Public Types

- typedef `_Base::allocator_type` **allocator_type**
- typedef `__gnu_debug::Safe_iterator<_Base_const_iterator, unordered_multimap>` **const_iterator**
- typedef `__gnu_debug::Safe_local_iterator<_Base_const_local_iterator, unordered_multimap>` **const_local_iterator**
- typedef `_Base::hasher` **hasher**
- typedef `__gnu_debug::Safe_iterator<_Base_iterator, unordered_multimap>` **iterator**
- typedef `_Base::key_equal` **key_equal**
- typedef `_Base::key_type` **key_type**
- typedef `__gnu_debug::Safe_local_iterator<_Base_local_iterator, unordered_multimap>` **local_iterator**
- typedef `_Base::size_type` **size_type**
- typedef `_Base::value_type` **value_type**

Public Member Functions

- **unordered_multimap** (size_type __n, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- template<typename _InputIterator >
unordered_multimap (_InputIterator __first, _InputIterator __last, size_type __n=0, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- **unordered_multimap** (const `unordered_multimap` &)=default
- **unordered_multimap** (const `_Base` &__x)
- **unordered_multimap** (`unordered_multimap` &&)=default
- **unordered_multimap** (const allocator_type &__a)
- **unordered_multimap** (const `unordered_multimap` &__umap, const allocator_type &__a)
- **unordered_multimap** (`unordered_multimap` &&__umap, const allocator_type &__a)

- **unordered_multimap** ([initializer_list](#)< value_type > __l, size_type __n=0, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- **unordered_multimap** (size_type __n, const allocator_type &__a)
- **unordered_multimap** (size_type __n, const hasher &__hf, const allocator_type &__a)
- template<typename _InputIterator >
unordered_multimap (_InputIterator __first, _InputIterator __last, size_type __n, const allocator_type &__a)
- template<typename _InputIterator >
unordered_multimap (_InputIterator __first, _InputIterator __last, size_type __n, const hasher &__hf, const allocator_type &__a)
- **unordered_multimap** ([initializer_list](#)< value_type > __l, size_type __n, const allocator_type &__a)
- **unordered_multimap** ([initializer_list](#)< value_type > __l, size_type __n, const hasher &__hf, const allocator_type &__a)
- void **_M_attach** (_Safe_iterator_base * __it, bool __constant)
- void **_M_attach_local** (_Safe_iterator_base * __it, bool __constant)
- void **_M_attach_local_single** (_Safe_iterator_base * __it, bool __constant) throw ()
- void **_M_attach_single** (_Safe_iterator_base * __it, bool __constant) throw ()
- **_Base** & **_M_base** () noexcept
- const **_Base** & **_M_base** () const noexcept
- void **_M_detach** (_Safe_iterator_base * __it)
- void **_M_detach_local** (_Safe_iterator_base * __it)
- void **_M_detach_local_single** (_Safe_iterator_base * __it) throw ()
- void **_M_detach_single** (_Safe_iterator_base * __it) throw ()
- void **_M_invalidate_all** () const
- void **_M_swap** (_Safe_container & __x) noexcept
- **iterator begin** () noexcept
- **const_iterator begin** () const noexcept
- **local_iterator begin** (size_type __b)
- **const_local_iterator begin** (size_type __b) const
- size_type **bucket_size** (size_type __b) const
- **const_iterator cbegin** () const noexcept
- **const_local_iterator cbegin** (size_type __b) const
- **const_iterator cend** () const noexcept
- **const_local_iterator cend** (size_type __b) const
- void **clear** () noexcept
- template<typename... _Args>
iterator emplace (_Args &&... __args)
- template<typename... _Args>
iterator emplace_hint (**const_iterator** __hint, _Args &&... __args)
- **iterator end** () noexcept
- **const_iterator end** () const noexcept
- **local_iterator end** (size_type __b)
- **const_local_iterator end** (size_type __b) const
- **std::pair**< **iterator**, **iterator** > **equal_range** (const key_type & __key)
- **std::pair**< **const_iterator**, **const_iterator** > **equal_range** (const key_type & __key) const
- size_type **erase** (const key_type & __key)
- **iterator erase** (**const_iterator** __it)
- **iterator erase** (**iterator** __it)
- **iterator erase** (**const_iterator** __first, **const_iterator** __last)
- **iterator find** (const key_type & __key)
- **const_iterator find** (const key_type & __key) const
- **iterator insert** (const value_type & __obj)

- [iterator insert](#) ([const_iterator](#) __hint, const value_type &__obj)
- [template](#)<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>
[iterator insert](#) (_Pair &&__obj)
- [template](#)<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>
[iterator insert](#) ([const_iterator](#) __hint, _Pair &&__obj)
- void [insert](#) ([std::initializer_list](#)< value_type > __l)
- [template](#)<typename _InputIterator >
void [insert](#) (_InputIterator __first, _InputIterator __last)
- float [max_load_factor](#) () const noexcept
- void [max_load_factor](#) (float __f)
- [unordered_multimap](#) & [operator=](#) (const [unordered_multimap](#) &)=default
- [unordered_multimap](#) & [operator=](#) ([unordered_multimap](#) &&)=default
- [unordered_multimap](#) & [operator=](#) ([initializer_list](#)< value_type > __l)
- void [swap](#) ([unordered_multimap](#) &__x) noexcept(noexcept(declval< [_Base](#) & >().swap(__x)))

Public Attributes

- [_Safe_iterator_base](#) * [_M_const_iterators](#)
- [_Safe_iterator_base](#) * [_M_const_local_iterators](#)
- [_Safe_iterator_base](#) * [_M_iterators](#)
- [_Safe_iterator_base](#) * [_M_local_iterators](#)
- unsigned int [_M_version](#)

Protected Member Functions

- void [_M_detach_all](#) ()
- void [_M_detach_singular](#) ()
- [__gnu_cxx::__mutex](#) & [_M_get_mutex](#) () throw ()
- void [_M_invalidate_all](#) ()
- void [_M_invalidate_if](#) (_Predicate __pred)
- void [_M_invalidate_local_if](#) (_Predicate __pred)
- void [_M_invalidate_locals](#) ()
- void [_M_revalidate_singular](#) ()
- [_Safe_container](#) & [_M_safe](#) () noexcept
- void [_M_swap](#) (_Safe_unordered_container_base &__x) noexcept
- void [_M_swap](#) (_Safe_sequence_base &__x) noexcept

5.44.1 Detailed Description

```
template<typename _Key, typename _Tp, typename _Hash = std::hash<_Key>, typename _Pred = std::equal_to<_Key>, typename
_Alloc = std::allocator<std::pair<const _Key, _Tp> >>
class std::__debug::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >
```

Class std::unordered_multimap with safety/checking/debug instrumentation.

Definition at line 594 of file debug/unordered_map.

5.444.2 Member Function Documentation

5.444.2.1 void __gnu_debug::Safe_sequence_base::M_attach (_Safe_iterator_base * __it, bool __constant)
[inherited]

Attach an iterator to this sequence.

5.444.2.2 void __gnu_debug::Safe_unordered_container_base::M_attach_local (_Safe_iterator_base * __it, bool __constant) [inherited]

Attach an iterator to this container.

5.444.2.3 void __gnu_debug::Safe_unordered_container_base::M_attach_local_single (_Safe_iterator_base * __it, bool __constant) throw) [inherited]

Likewise but not thread safe.

5.444.2.4 void __gnu_debug::Safe_sequence_base::M_attach_single (_Safe_iterator_base * __it, bool __constant) throw) [inherited]

Likewise but not thread safe.

5.444.2.5 void __gnu_debug::Safe_sequence_base::M_detach (_Safe_iterator_base * __it) [inherited]

Detach an iterator from this sequence

Referenced by __gnu_debug::Safe_iterator< _Iterator, _Sequence >::Safe_iterator().

5.444.2.6 void __gnu_debug::Safe_unordered_container_base::M_detach_all () [protected], [inherited]

Detach all iterators, leaving them singular.

5.444.2.7 void __gnu_debug::Safe_unordered_container_base::M_detach_local (_Safe_iterator_base * __it) [inherited]

Detach an iterator from this container

5.444.2.8 void __gnu_debug::Safe_unordered_container_base::M_detach_local_single (_Safe_iterator_base * __it) throw) [inherited]

Likewise but not thread safe.

5.444.2.9 void __gnu_debug::Safe_sequence_base::M_detach_single (_Safe_iterator_base * __it) throw) [inherited]

Likewise but not thread safe.

5.444.2.10 `void __gnu_debug::Safe_sequence_base::M_detach_singular ()` [protected],[inherited]

Detach all singular iterators.

Postcondition

for all iterators *i* attached to this sequence, *i*->_M_version == _M_version.

5.444.2.11 `__gnu_cxx::mutex& __gnu_debug::Safe_sequence_base::M_get_mutex () throw ()` [protected],[inherited]

For use in _Safe_sequence.

Referenced by __gnu_debug::Safe_sequence<_Sequence>::M_transfer_from_if().

5.444.2.12 `void __gnu_debug::Safe_sequence_base::M_invalidate_all () const` [inline],[inherited]

Invalidates all iterators.

Definition at line 248 of file safe_base.h.

References __gnu_debug::Safe_iterator_base::M_attach(), __gnu_debug::Safe_iterator_base::M_attach_single(), __gnu_debug::Safe_iterator_base::M_detach(), and __gnu_debug::Safe_iterator_base::M_detach_single().

5.444.2.13 `void __gnu_debug::Safe_unordered_container<unordered_multimap<_Key,_Tp,_Hash,_Pred,_Alloc>>::M_invalidate_if (_Predicate __pred)` [protected],[inherited]

Invalidates all iterators *x* that reference this container, are not singular, and for which __pred(*x*) returns true. __pred will be invoked with the normal iterators nested in the safe ones.

5.444.2.14 `void __gnu_debug::Safe_unordered_container<unordered_multimap<_Key,_Tp,_Hash,_Pred,_Alloc>>::M_invalidate_local_if (_Predicate __pred)` [protected],[inherited]

Invalidates all local iterators *x* that reference this container, are not singular, and for which __pred(*x*) returns true. __pred will be invoked with the normal ilocal iterators nested in the safe ones.

5.444.2.15 `void __gnu_debug::Safe_sequence_base::M_revalidate_singular ()` [protected],[inherited]

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

5.444.2.16 `void __gnu_debug::Safe_unordered_container_base::M_swap (_Safe_unordered_container_base & __x)` [protected],[noexcept],[inherited]

Swap this container with the given container. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.444.2.17 void __gnu_debug::Safe_sequence_base::M_swap (_Safe_sequence_base & __x) [protected],
[noexcept],[inherited]

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.444.3 Member Data Documentation

5.444.3.1 _Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_const_iterators [inherited]

The list of constant iterators that reference this container.

Definition at line 193 of file safe_base.h.

Referenced by __gnu_debug::Safe_sequence<_Sequence>::M_transfer_from_if().

5.444.3.2 _Safe_iterator_base* __gnu_debug::Safe_unordered_container_base::M_const_local_iterators [inherited]

The list of constant local iterators that reference this container.

Definition at line 125 of file safe_unordered_base.h.

5.444.3.3 _Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_iterators [inherited]

The list of mutable iterators that reference this container.

Definition at line 190 of file safe_base.h.

Referenced by __gnu_debug::Safe_sequence<_Sequence>::M_transfer_from_if().

5.444.3.4 _Safe_iterator_base* __gnu_debug::Safe_unordered_container_base::M_local_iterators [inherited]

The list of mutable local iterators that reference this container.

Definition at line 122 of file safe_unordered_base.h.

5.444.3.5 unsigned int __gnu_debug::Safe_sequence_base::M_version [mutable],[inherited]

The container version number. This number may never be 0.

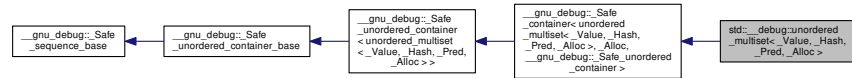
Definition at line 196 of file safe_base.h.

The documentation for this class was generated from the following file:

- [debug/unordered_map](#)

5.445 `std::__debug::unordered_multiset<_Value,_Hash,_Pred,_Alloc>` Class Template Reference

Inheritance diagram for `std::__debug::unordered_multiset<_Value,_Hash,_Pred,_Alloc>`:



Public Types

- typedef `_Base::allocator_type` **allocator_type**
- typedef `__gnu_debug::Safe_iterator<_Base_const_iterator, unordered_multiset>` **const_iterator**
- typedef `__gnu_debug::Safe_local_iterator<_Base_const_local_iterator, unordered_multiset>` **const_local_iterator**
- typedef `_Base::hasher` **hasher**
- typedef `__gnu_debug::Safe_iterator<_Base_iterator, unordered_multiset>` **iterator**
- typedef `_Base::key_equal` **key_equal**
- typedef `_Base::key_type` **key_type**
- typedef `__gnu_debug::Safe_local_iterator<_Base_local_iterator, unordered_multiset>` **local_iterator**
- typedef `_Base::size_type` **size_type**
- typedef `_Base::value_type` **value_type**

Public Member Functions

- **unordered_multiset** (`size_type __n`, `const hasher &__hf=hasher()`, `const key_equal &__eq=key_equal()`, `const allocator_type &__a=allocator_type()`)
- `template<typename _InputIterator>`
unordered_multiset (`_InputIterator __first`, `_InputIterator __last`, `size_type __n=0`, `const hasher &__hf=hasher()`, `const key_equal &__eq=key_equal()`, `const allocator_type &__a=allocator_type()`)
- **unordered_multiset** (`const unordered_multiset &`)=default
- **unordered_multiset** (`const _Base &x`)
- **unordered_multiset** (`unordered_multiset &&`)=default
- **unordered_multiset** (`const allocator_type &a`)
- **unordered_multiset** (`const unordered_multiset &__uset`, `const allocator_type &a`)
- **unordered_multiset** (`unordered_multiset &&__uset`, `const allocator_type &a`)
- **unordered_multiset** (`initializer_list<value_type> __l`, `size_type __n=0`, `const hasher &__hf=hasher()`, `const key_equal &__eq=key_equal()`, `const allocator_type &a=allocator_type()`)
- **unordered_multiset** (`size_type __n`, `const allocator_type &a`)
- **unordered_multiset** (`size_type __n`, `const hasher &__hf`, `const allocator_type &a`)
- `template<typename _InputIterator>`
unordered_multiset (`_InputIterator __first`, `_InputIterator __last`, `size_type __n`, `const allocator_type &a`)
- `template<typename _InputIterator>`
unordered_multiset (`_InputIterator __first`, `_InputIterator __last`, `size_type __n`, `const hasher &__hf`, `const allocator_type &a`)
- **unordered_multiset** (`initializer_list<value_type> __l`, `size_type __n`, `const allocator_type &a`)

- `unordered_multiset` (`initializer_list`< `value_type` > `__l`, `size_type` `__n`, `const hasher` & `__hf`, `const allocator_type` & `__a`)
- `void __M_attach` (`_Safe_iterator_base` * `__it`, `bool` `__constant`)
- `void __M_attach_local` (`_Safe_iterator_base` * `__it`, `bool` `__constant`)
- `void __M_attach_local_single` (`_Safe_iterator_base` * `__it`, `bool` `__constant`) `throw ()`
- `void __M_attach_single` (`_Safe_iterator_base` * `__it`, `bool` `__constant`) `throw ()`
- `__Base` & `__M_base` () `noexcept`
- `const __Base` & `__M_base` () `const noexcept`
- `void __M_detach` (`_Safe_iterator_base` * `__it`)
- `void __M_detach_local` (`_Safe_iterator_base` * `__it`)
- `void __M_detach_local_single` (`_Safe_iterator_base` * `__it`) `throw ()`
- `void __M_detach_single` (`_Safe_iterator_base` * `__it`) `throw ()`
- `void __M_invalidate_all` () `const`
- `void __M_swap` (`_Safe_container` & `__x`) `noexcept`
- `iterator begin` () `noexcept`
- `const_iterator begin` () `const noexcept`
- `local_iterator begin` (`size_type` `__b`)
- `const_local_iterator begin` (`size_type` `__b`) `const`
- `size_type bucket_size` (`size_type` `__b`) `const`
- `const_iterator cbegin` () `const noexcept`
- `const_local_iterator cbegin` (`size_type` `__b`) `const`
- `const_iterator cend` () `const noexcept`
- `const_local_iterator cend` (`size_type` `__b`) `const`
- `void clear` () `noexcept`
- `template<typename... _Args>`
`iterator emplace` (`_Args` &&... `__args`)
- `template<typename... _Args>`
`iterator emplace_hint` (`const_iterator` `__hint`, `_Args` &&... `__args`)
- `iterator end` () `noexcept`
- `const_iterator end` () `const noexcept`
- `local_iterator end` (`size_type` `__b`)
- `const_local_iterator end` (`size_type` `__b`) `const`
- `std::pair< iterator, iterator > equal_range` (`const key_type` & `__key`)
- `std::pair< const_iterator, const_iterator > equal_range` (`const key_type` & `__key`) `const`
- `size_type erase` (`const key_type` & `__key`)
- `iterator erase` (`const_iterator` `__it`)
- `iterator erase` (`iterator` `__it`)
- `iterator erase` (`const_iterator` `__first`, `const_iterator` `__last`)
- `iterator find` (`const key_type` & `__key`)
- `const_iterator find` (`const key_type` & `__key`) `const`
- `iterator insert` (`const value_type` & `__obj`)
- `iterator insert` (`const_iterator` `__hint`, `const value_type` & `__obj`)
- `iterator insert` (`value_type` && `__obj`)
- `iterator insert` (`const_iterator` `__hint`, `value_type` && `__obj`)
- `void insert` (`std::initializer_list`< `value_type` > `__l`)
- `template<typename _InputIterator >`
`void insert` (`_InputIterator` `__first`, `_InputIterator` `__last`)
- `float max_load_factor` () `const noexcept`
- `void max_load_factor` (`float` `__f`)
- `unordered_multiset` & `operator=` (`const unordered_multiset` &) = default
- `unordered_multiset` & `operator=` (`unordered_multiset` &&) = default
- `unordered_multiset` & `operator=` (`initializer_list`< `value_type` > `__l`)
- `void swap` (`unordered_multiset` & `__x`) `noexcept(noexcept(declval< __Base >().swap(__x)))`

Public Attributes

- `_Safe_iterator_base * _M_const_iterators`
- `_Safe_iterator_base * _M_const_local_iterators`
- `_Safe_iterator_base * _M_iterators`
- `_Safe_iterator_base * _M_local_iterators`
- `unsigned int _M_version`

Protected Member Functions

- `void _M_detach_all ()`
- `void _M_detach_singular ()`
- `__gnu_cxx::__mutex & _M_get_mutex () throw ()`
- `void _M_invalidate_all ()`
- `void _M_invalidate_if (_Predicate __pred)`
- `void _M_invalidate_local_if (_Predicate __pred)`
- `void _M_invalidate_locals ()`
- `void _M_revalidate_singular ()`
- `_Safe_container & _M_safe () noexcept`
- `void _M_swap (_Safe_unordered_container_base &__x) noexcept`
- `void _M_swap (_Safe_sequence_base &__x) noexcept`

5.445.1 Detailed Description

```
template<typename _Value, typename _Hash = std::hash<_Value>, typename _Pred = std::equal_to<_Value>, typename _Alloc =
std::allocator<_Value>>
class std::__debug::unordered_multiset< _Value, _Hash, _Pred, _Alloc >
```

Class `std::unordered_multiset` with safety/checking/debug instrumentation.

Definition at line 504 of file `debug/unordered_set`.

5.445.2 Member Function Documentation

5.445.2.1 `void __gnu_debug::Safe_sequence_base::_M_attach (_Safe_iterator_base * __it, bool __constant)`
[inherited]

Attach an iterator to this sequence.

5.445.2.2 `void __gnu_debug::Safe_unordered_container_base::_M_attach_local (_Safe_iterator_base * __it, bool __constant)`
[inherited]

Attach an iterator to this container.

5.445.2.3 void __gnu_debug::Safe_unordered_container_base::M_attach_local_single (_Safe_iterator_base * __it, bool __constant) throw) [inherited]

Likewise but not thread safe.

5.445.2.4 void __gnu_debug::Safe_sequence_base::M_attach_single (_Safe_iterator_base * __it, bool __constant) throw) [inherited]

Likewise but not thread safe.

5.445.2.5 void __gnu_debug::Safe_sequence_base::M_detach (_Safe_iterator_base * __it) [inherited]

Detach an iterator from this sequence

Referenced by __gnu_debug::Safe_iterator< _Iterator, _Sequence >::Safe_iterator().

5.445.2.6 void __gnu_debug::Safe_unordered_container_base::M_detach_all () [protected],[inherited]

Detach all iterators, leaving them singular.

5.445.2.7 void __gnu_debug::Safe_unordered_container_base::M_detach_local (_Safe_iterator_base * __it) [inherited]

Detach an iterator from this container

5.445.2.8 void __gnu_debug::Safe_unordered_container_base::M_detach_local_single (_Safe_iterator_base * __it) throw) [inherited]

Likewise but not thread safe.

5.445.2.9 void __gnu_debug::Safe_sequence_base::M_detach_single (_Safe_iterator_base * __it) throw) [inherited]

Likewise but not thread safe.

5.445.2.10 void __gnu_debug::Safe_sequence_base::M_detach_singular () [protected],[inherited]

Detach all singular iterators.

Postcondition

for all iterators i attached to this sequence, i->_M_version == _M_version.

5.445.2.11 __gnu_cxx::mutex& __gnu_debug::Safe_sequence_base::M_get_mutex () throw) [protected],[inherited]

For use in _Safe_sequence.

Referenced by __gnu_debug::Safe_sequence< _Sequence >::M_transfer_from_if().

5.445.2.12 `void __gnu_debug::Safe_sequence_base::M_invalidate_all() const` `[inline],[inherited]`

Invalidates all iterators.

Definition at line 248 of file `safe_base.h`.

References `__gnu_debug::Safe_iterator_base::M_attach()`, `__gnu_debug::Safe_iterator_base::M_attach_single()`, `__gnu_debug::Safe_iterator_base::M_detach()`, and `__gnu_debug::Safe_iterator_base::M_detach_single()`.

5.445.2.13 `void __gnu_debug::Safe_unordered_container<unordered_multiset<_Value,_Hash,_Pred,_Alloc>
>::M_invalidate_if(_Predicate __pred)` `[protected],[inherited]`

Invalidates all iterators `x` that reference this container, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal iterators nested in the safe ones.

5.445.2.14 `void __gnu_debug::Safe_unordered_container<unordered_multiset<_Value,_Hash,_Pred,_Alloc>
>::M_invalidate_local_if(_Predicate __pred)` `[protected],[inherited]`

Invalidates all local iterators `x` that reference this container, are not singular, and for which `__pred(x)` returns true. `__pred` will be invoked with the normal ilocal iterators nested in the safe ones.

5.445.2.15 `void __gnu_debug::Safe_sequence_base::M_revalidate_singular()` `[protected],[inherited]`

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

5.445.2.16 `void __gnu_debug::Safe_unordered_container_base::M_swap(_Safe_unordered_container_base &__x)` `[protected],[noexcept],[inherited]`

Swap this container with the given container. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.445.2.17 `void __gnu_debug::Safe_sequence_base::M_swap(_Safe_sequence_base &__x)` `[protected],[noexcept],[inherited]`

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.445.3 Member Data Documentation

5.445.3.1 `_Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_const_iterators` `[inherited]`

The list of constant iterators that reference this container.

Definition at line 193 of file `safe_base.h`.

Referenced by `__gnu_debug::Safe_sequence<_Sequence>::M_transfer_from_if()`.

5.445.3.2 _Safe_iterator_base* __gnu_debug::Safe_unordered_container_base::M_const_local_iterators [inherited]

The list of constant local iterators that reference this container.

Definition at line 125 of file safe_unordered_base.h.

5.445.3.3 _Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_iterators [inherited]

The list of mutable iterators that reference this container.

Definition at line 190 of file safe_base.h.

Referenced by __gnu_debug::Safe_sequence< _Sequence >::M_transfer_from_if().

5.445.3.4 _Safe_iterator_base* __gnu_debug::Safe_unordered_container_base::M_local_iterators [inherited]

The list of mutable local iterators that reference this container.

Definition at line 122 of file safe_unordered_base.h.

5.445.3.5 unsigned int __gnu_debug::Safe_sequence_base::M_version [mutable], [inherited]

The container version number. This number may never be 0.

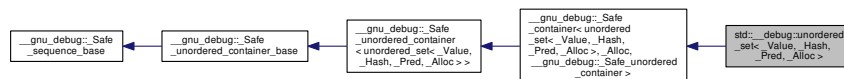
Definition at line 196 of file safe_base.h.

The documentation for this class was generated from the following file:

- [debug/unordered_set](#)

5.446 std::__debug::unordered_set< _Value, _Hash, _Pred, _Alloc > Class Template Reference

Inheritance diagram for std::__debug::unordered_set< _Value, _Hash, _Pred, _Alloc >:



Public Types

- typedef _Base::allocator_type **allocator_type**
- typedef __gnu_debug::Safe_iterator< _Base_const_iterator, unordered_set > **const_iterator**
- typedef __gnu_debug::Safe_local_iterator< _Base_const_local_iterator, unordered_set > **const_local_iterator**
- typedef _Base::hasher **hasher**
- typedef __gnu_debug::Safe_iterator< _Base_iterator, unordered_set > **iterator**
- typedef _Base::key_equal **key_equal**
- typedef _Base::key_type **key_type**
- typedef __gnu_debug::Safe_local_iterator< _Base_local_iterator, unordered_set > **local_iterator**
- typedef _Base::size_type **size_type**
- typedef _Base::value_type **value_type**

Public Member Functions

- **unordered_set** (size_type __n, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- template<typename _InputIterator >
unordered_set (_InputIterator __first, _InputIterator __last, size_type __n=0, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- **unordered_set** (const [unordered_set](#) &)=default
- **unordered_set** (const [_Base](#) &__x)
- **unordered_set** ([unordered_set](#) &&)=default
- **unordered_set** (const allocator_type &__a)
- **unordered_set** (const [unordered_set](#) &__uset, const allocator_type &__a)
- **unordered_set** ([unordered_set](#) && __uset, const allocator_type &__a)
- **unordered_set** ([initializer_list](#)< value_type > __l, size_type __n=0, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- **unordered_set** (size_type __n, const allocator_type &__a)
- **unordered_set** (size_type __n, const hasher &__hf, const allocator_type &__a)
- template<typename _InputIterator >
unordered_set (_InputIterator __first, _InputIterator __last, size_type __n, const allocator_type &__a)
- template<typename _InputIterator >
unordered_set (_InputIterator __first, _InputIterator __last, size_type __n, const hasher &__hf, const allocator_type &__a)
- **unordered_set** ([initializer_list](#)< value_type > __l, size_type __n, const allocator_type &__a)
- **unordered_set** ([initializer_list](#)< value_type > __l, size_type __n, const hasher &__hf, const allocator_type &__a)
- void [_M_attach](#) (_Safe_iterator_base *__it, bool __constant)
- void [_M_attach_local](#) (_Safe_iterator_base *__it, bool __constant)
- void [_M_attach_local_single](#) (_Safe_iterator_base *__it, bool __constant) throw ()
- void [_M_attach_single](#) (_Safe_iterator_base *__it, bool __constant) throw ()
- [_Base](#) & [_M_base](#) () noexcept
- const [_Base](#) & [_M_base](#) () const noexcept
- void [_M_detach](#) (_Safe_iterator_base *__it)
- void [_M_detach_local](#) (_Safe_iterator_base *__it)
- void [_M_detach_local_single](#) (_Safe_iterator_base *__it) throw ()
- void [_M_detach_single](#) (_Safe_iterator_base *__it) throw ()
- void [_M_invalidate_all](#) () const
- void [_M_swap](#) (_Safe_container &__x) noexcept
- [iterator begin](#) () noexcept
- [const_iterator begin](#) () const noexcept
- [local_iterator begin](#) (size_type __b)
- [const_local_iterator begin](#) (size_type __b) const
- size_type [bucket_size](#) (size_type __b) const
- [const_iterator cbegin](#) () const noexcept
- [const_local_iterator cbegin](#) (size_type __b) const
- [const_iterator cend](#) () const noexcept
- [const_local_iterator cend](#) (size_type __b) const
- void [clear](#) () noexcept
- template<typename... _Args>
[std::pair](#)< [iterator](#), bool > [emplace](#) (_Args &&... __args)
- template<typename... _Args>
[iterator emplace_hint](#) ([const_iterator](#) __hint, _Args &&... __args)
- [iterator end](#) () noexcept

- `const_iterator end` () const noexcept
- `local_iterator end` (size_type __b)
- `const_local_iterator end` (size_type __b) const
- `std::pair< iterator, iterator > equal_range` (const key_type &__key)
- `std::pair< const_iterator, const_iterator > equal_range` (const key_type &__key) const
- size_type `erase` (const key_type &__key)
- `iterator erase` (const_iterator __it)
- `iterator erase` (iterator __it)
- `iterator erase` (const_iterator __first, const_iterator __last)
- `iterator find` (const key_type &__key)
- `const_iterator find` (const key_type &__key) const
- `std::pair< iterator, bool > insert` (const value_type &__obj)
- `iterator insert` (const_iterator __hint, const value_type &__obj)
- `std::pair< iterator, bool > insert` (value_type &&__obj)
- `iterator insert` (const_iterator __hint, value_type &&__obj)
- void `insert` (std::initializer_list< value_type > __l)
- template<typename _InputIterator >
void `insert` (_InputIterator __first, _InputIterator __last)
- float `max_load_factor` () const noexcept
- void `max_load_factor` (float __f)
- `unordered_set & operator=` (const `unordered_set` &)=default
- `unordered_set & operator=` (`unordered_set` &&)=default
- `unordered_set & operator=` (initializer_list< value_type > __l)
- void `swap` (`unordered_set` &__x) noexcept(noexcept(declval<_Base &>().swap(__x)))

Public Attributes

- _Safe_iterator_base * `_M_const_iterators`
- _Safe_iterator_base * `_M_const_local_iterators`
- _Safe_iterator_base * `_M_iterators`
- _Safe_iterator_base * `_M_local_iterators`
- unsigned int `_M_version`

Protected Member Functions

- void `_M_detach_all` ()
- void `_M_detach_singular` ()
- __gnu_cxx::__mutex & `_M_get_mutex` () throw ()
- void `_M_invalidate_all` ()
- void `_M_invalidate_if` (_Predicate __pred)
- void `_M_invalidate_local_if` (_Predicate __pred)
- void `_M_invalidate_locals` ()
- void `_M_revalidate_singular` ()
- _Safe_container & `_M_safe` () noexcept
- void `_M_swap` (_Safe_unordered_container_base &__x) noexcept
- void `_M_swap` (_Safe_sequence_base &__x) noexcept

5.446.1 Detailed Description

```
template<typename _Value, typename _Hash = std::hash<_Value>, typename _Pred = std::equal_to<_Value>, typename _Alloc =
std::allocator<_Value>>
class std::__debug::unordered_set< _Value, _Hash, _Pred, _Alloc >
```

Class std::unordered_set with safety/checking/debug instrumentation.

Definition at line 51 of file debug/unordered_set.

5.446.2 Member Function Documentation

```
5.446.2.1 void __gnu_debug::Safe_sequence_base::_M_attach ( _Safe_iterator_base * __it, bool __constant )
[inherited]
```

Attach an iterator to this sequence.

```
5.446.2.2 void __gnu_debug::Safe_unordered_container_base::_M_attach_local ( _Safe_iterator_base * __it, bool __constant
) [inherited]
```

Attach an iterator to this container.

```
5.446.2.3 void __gnu_debug::Safe_unordered_container_base::_M_attach_local_single ( _Safe_iterator_base * __it, bool
__constant ) throw ) [inherited]
```

Likewise but not thread safe.

```
5.446.2.4 void __gnu_debug::Safe_sequence_base::_M_attach_single ( _Safe_iterator_base * __it, bool __constant ) throw )
[inherited]
```

Likewise but not thread safe.

```
5.446.2.5 void __gnu_debug::Safe_sequence_base::_M_detach ( _Safe_iterator_base * __it ) [inherited]
```

Detach an iterator from this sequence

Referenced by __gnu_debug::Safe_iterator< _Iterator, _Sequence >::_Safe_iterator().

```
5.446.2.6 void __gnu_debug::Safe_unordered_container_base::_M_detach_all ( ) [protected],[inherited]
```

Detach all iterators, leaving them singular.

```
5.446.2.7 void __gnu_debug::Safe_unordered_container_base::_M_detach_local ( _Safe_iterator_base * __it )
[inherited]
```

Detach an iterator from this container

5.446.2.8 void __gnu_debug::Safe_unordered_container_base::M_detach_local_single (_Safe_iterator_base * __it) throw (
 [inherited]

Likewise but not thread safe.

5.446.2.9 void __gnu_debug::Safe_sequence_base::M_detach_single (_Safe_iterator_base * __it) throw (
 [inherited]

Likewise but not thread safe.

5.446.2.10 void __gnu_debug::Safe_sequence_base::M_detach_singular () [protected], [inherited]

Detach all singular iterators.

Postcondition

for all iterators i attached to this sequence, i->_M_version == _M_version.

5.446.2.11 __gnu_cxx::mutex& __gnu_debug::Safe_sequence_base::M_get_mutex () throw ([protected],
 [inherited]

For use in _Safe_sequence.

Referenced by __gnu_debug::Safe_sequence<_Sequence >::M_transfer_from_if().

5.446.2.12 void __gnu_debug::Safe_sequence_base::M_invalidate_all () const [inline], [inherited]

Invalidates all iterators.

Definition at line 248 of file safe_base.h.

References __gnu_debug::Safe_iterator_base::M_attach(), __gnu_debug::Safe_iterator_base::M_attach_single(),
 __gnu_debug::Safe_iterator_base::M_detach(), and __gnu_debug::Safe_iterator_base::M_detach_single().

5.446.2.13 void __gnu_debug::Safe_unordered_container<unordered_set<_Value, _Hash, _Pred, _Alloc >
 >::M_invalidate_if (_Predicate __pred) [protected], [inherited]

Invalidates all iterators x that reference this container, are not singular, and for which __pred(x) returns true.
 __pred will be invoked with the normal iterators nested in the safe ones.

5.446.2.14 void __gnu_debug::Safe_unordered_container<unordered_set<_Value, _Hash, _Pred, _Alloc >
 >::M_invalidate_local_if (_Predicate __pred) [protected], [inherited]

Invalidates all local iterators x that reference this container, are not singular, and for which __pred(x) returns true.
 __pred will be invoked with the normal ilocal iterators nested in the safe ones.

5.446.2.15 `void __gnu_debug::Safe_sequence_base::M_revalidate_singular ()` [protected],[inherited]

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

5.446.2.16 `void __gnu_debug::Safe_unordered_container_base::M_swap (_Safe_unordered_container_base & __x)` [protected],[noexcept],[inherited]

Swap this container with the given container. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.446.2.17 `void __gnu_debug::Safe_sequence_base::M_swap (_Safe_sequence_base & __x)` [protected],[noexcept],[inherited]

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.446.3 Member Data Documentation

5.446.3.1 `_Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_const_iterators` [inherited]

The list of constant iterators that reference this container.

Definition at line 193 of file `safe_base.h`.

Referenced by `__gnu_debug::Safe_sequence< _Sequence >::M_transfer_from_if()`.

5.446.3.2 `_Safe_iterator_base* __gnu_debug::Safe_unordered_container_base::M_const_local_iterators` [inherited]

The list of constant local iterators that reference this container.

Definition at line 125 of file `safe_unordered_base.h`.

5.446.3.3 `_Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_iterators` [inherited]

The list of mutable iterators that reference this container.

Definition at line 190 of file `safe_base.h`.

Referenced by `__gnu_debug::Safe_sequence< _Sequence >::M_transfer_from_if()`.

5.446.3.4 `_Safe_iterator_base* __gnu_debug::Safe_unordered_container_base::M_local_iterators` [inherited]

The list of mutable local iterators that reference this container.

Definition at line 122 of file `safe_unordered_base.h`.

5.446.3.5 unsigned int __gnu_debug::Safe_sequence_base::_M_version [mutable],[inherited]

The container version number. This number may never be 0.

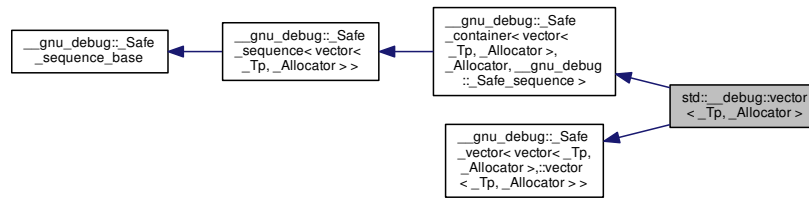
Definition at line 196 of file safe_base.h.

The documentation for this class was generated from the following file:

- [debug/unordered_set](#)

5.447 std::__debug::vector< _Tp, _Allocator > Class Template Reference

Inheritance diagram for std::__debug::vector< _Tp, _Allocator >:



Public Types

- typedef `_Allocator` **allocator_type**
- typedef `__gnu_debug::Safe_iterator< _Base_const_iterator, vector >` **const_iterator**
- typedef `_Base::const_pointer` **const_pointer**
- typedef `_Base::const_reference` **const_reference**
- typedef `std::reverse_iterator< const_iterator >` **const_reverse_iterator**
- typedef `_Base::difference_type` **difference_type**
- typedef `__gnu_debug::Safe_iterator< _Base_iterator, vector >` **iterator**
- typedef `_Base::pointer` **pointer**
- typedef `_Base::reference` **reference**
- typedef `std::reverse_iterator< iterator >` **reverse_iterator**
- typedef `_Base::size_type` **size_type**
- typedef `_Tp` **value_type**

Public Member Functions

- **vector** (const `_Allocator` &__a) noexcept
- **vector** (size_type __n, const `_Allocator` &__a= `_Allocator`())
- **vector** (size_type __n, const `_Tp` &__value, const `_Allocator` &__a= `_Allocator`())
- template<class `_InputIterator` , typename = std::RequireInputIter<`_InputIterator`>>
 vector (`_InputIterator` __first, `_InputIterator` __last, const `_Allocator` &__a= `_Allocator`())
- **vector** (const `vector` &)=default
- **vector** (`vector` &&)=default
- **vector** (const `vector` &__x, const allocator_type &__a)
- **vector** (`vector` &&__x, const allocator_type &__a)
- **vector** (initializer_list< value_type > __l, const allocator_type &__a=allocator_type())
- `vector` (const `_Base` &__x)
- void `_M_attach` (`_Safe_iterator_base` *__it, bool __constant)
- void `_M_attach_single` (`_Safe_iterator_base` *__it, bool __constant) throw ()
- `_Base` & `_M_base` () noexcept
- const `_Base` & `_M_base` () const noexcept
- void `_M_detach` (`_Safe_iterator_base` *__it)
- void `_M_detach_single` (`_Safe_iterator_base` *__it) throw ()
- void `_M_invalidate_all` () const
- void `_M_invalidate_if` (`_Predicate` __pred)
- void `_M_swap` (`_Safe_container` &__x) noexcept
- void `_M_transfer_from_if` (`_Safe_sequence` &__from, `_Predicate` __pred)
- template<typename `_InputIterator` , typename = std::RequireInputIter<`_InputIterator`>>
 void **assign** (`_InputIterator` __first, `_InputIterator` __last)
- void **assign** (size_type __n, const `_Tp` &__u)
- void **assign** (initializer_list< value_type > __l)
- reference **back** () noexcept
- const_reference **back** () const noexcept
- `iterator` **begin** () noexcept
- const_iterator **begin** () const noexcept
- size_type **capacity** () const noexcept
- const_iterator **cbegin** () const noexcept
- const_iterator **cend** () const noexcept
- void **clear** () noexcept
- const_reverse_iterator **crbegin** () const noexcept
- const_reverse_iterator **crend** () const noexcept
- template<typename... `_Args`>
 `iterator` **emplace** (const_iterator __position, `_Args` &&... __args)
- template<typename... `_Args`>
 void **emplace_back** (`_Args` &&... __args)
- `iterator` **end** () noexcept
- const_iterator **end** () const noexcept
- `iterator` **erase** (const_iterator __position)
- `iterator` **erase** (const_iterator __first, const_iterator __last)
- reference **front** () noexcept
- const_reference **front** () const noexcept
- `iterator` **insert** (const_iterator __position, const `_Tp` &__x)
- template<typename `_Up` = `_Tp`>
 __gnu_cxx::enable_if<!std::are_same< `_Up`, bool >::value, `iterator` >::__type **insert** (const_iterator __position, `_Tp` &&__x)

- **iterator insert** ([const_iterator](#) __position, [initializer_list](#)< value_type > __l)
- **iterator insert** ([const_iterator](#) __position, size_type __n, const _Tp &__x)
- **template**<class _InputIterator, typename = std::__RequireInputIter<_InputIterator>>
iterator insert ([const_iterator](#) __position, _InputIterator __first, _InputIterator __last)
- **vector & operator=** (const [vector](#) &)=default
- **vector & operator=** ([vector](#) &&)=default
- **vector & operator=** ([initializer_list](#)< value_type > __l)
- **reference operator[]** (size_type __n) noexcept
- **const_reference operator[]** (size_type __n) const noexcept
- **void pop_back** () noexcept
- **void push_back** (const _Tp &__x)
- **template**<typename _Up = _Tp>
__gnu_cxx::__enable_if<!std::__are_same<_Up, bool>::__value, void>::__type **push_back** (_Tp &&__x)
- **reverse_iterator rbegin** () noexcept
- **const_reverse_iterator rbegin** () const noexcept
- **reverse_iterator rend** () noexcept
- **const_reverse_iterator rend** () const noexcept
- **void reserve** (size_type __n)
- **void resize** (size_type __sz)
- **void resize** (size_type __sz, const _Tp &__c)
- **void shrink_to_fit** ()
- **void swap** ([vector](#) &__x) noexcept(*/*conditional */*)

Public Attributes

- [_Safe_iterator_base](#) * [_M_const_iterators](#)
- [_Safe_iterator_base](#) * [_M_iterators](#)
- unsigned int [_M_version](#)

Protected Member Functions

- **void** [_M_detach_all](#) ()
- **void** [_M_detach_singular](#) ()
- [__gnu_cxx::__mutex](#) & [_M_get_mutex](#) () throw ()
- **bool** [_M_requires_reallocation](#) (size_type __elements) const noexcept
- **void** [_M_revalidate_singular](#) ()
- [_Safe_container](#) & [_M_safe](#) () noexcept
- **void** [_M_swap](#) ([_Safe_sequence_base](#) &__x) noexcept
- **void** [_M_update_guaranteed_capacity](#) () noexcept

Protected Attributes

- size_type [_M_guaranteed_capacity](#)

5.447.1 Detailed Description

```
template<typename _Tp, typename _Allocator = std::allocator<_Tp>>
class std::__debug::vector< _Tp, _Allocator >
```

Class std::vector with safety/checking/debug instrumentation.

Definition at line 111 of file debug/vector.

5.447.2 Constructor & Destructor Documentation

```
5.447.2.1 template<typename _Tp, typename _Allocator = std::allocator<_Tp>> std::__debug::vector< _Tp, _Allocator
>::vector ( const _Base & __x ) [inline]
```

Construction from a normal-mode vector.

Definition at line 212 of file debug/vector.

5.447.3 Member Function Documentation

```
5.447.3.1 void __gnu_debug::Safe_sequence_base::M_attach ( _Safe_iterator_base * __it, bool __constant )
[inherited]
```

Attach an iterator to this sequence.

```
5.447.3.2 void __gnu_debug::Safe_sequence_base::M_attach_single ( _Safe_iterator_base * __it, bool __constant ) throw )
[inherited]
```

Likewise but not thread safe.

```
5.447.3.3 void __gnu_debug::Safe_sequence_base::M_detach ( _Safe_iterator_base * __it ) [inherited]
```

Detach an iterator from this sequence

Referenced by __gnu_debug::Safe_iterator< _Iterator, _Sequence >::Safe_iterator().

```
5.447.3.4 void __gnu_debug::Safe_sequence_base::M_detach_all ( ) [protected],[inherited]
```

Detach all iterators, leaving them singular.

```
5.447.3.5 void __gnu_debug::Safe_sequence_base::M_detach_single ( _Safe_iterator_base * __it ) throw )
[inherited]
```

Likewise but not thread safe.

5.447.3.6 void __gnu_debug::Safe_sequence_base::M_detach_singular () [protected],[inherited]

Detach all singular iterators.

Postcondition

for all iterators *i* attached to this sequence, *i*->_M_version == _M_version.

5.447.3.7 __gnu_cxx::mutex& __gnu_debug::Safe_sequence_base::M_get_mutex () throw () [protected],[inherited]

For use in _Safe_sequence.

Referenced by __gnu_debug::Safe_sequence<_Sequence >::M_transfer_from_if().

5.447.3.8 void __gnu_debug::Safe_sequence_base::M_invalidate_all () const [inline],[inherited]

Invalidates all iterators.

Definition at line 248 of file safe_base.h.

References __gnu_debug::Safe_iterator_base::M_attach(), __gnu_debug::Safe_iterator_base::M_attach_single(), __gnu_debug::Safe_iterator_base::M_detach(), and __gnu_debug::Safe_iterator_base::M_detach_single().

5.447.3.9 void __gnu_debug::Safe_sequence<vector<_Tp, _Allocator > >::M_invalidate_if (_Predicate __pred) [inherited]

Invalidates all iterators *x* that reference this sequence, are not singular, and for which __pred(*x*) returns true. __pred will be invoked with the normal iterators nested in the safe ones.

5.447.3.10 void __gnu_debug::Safe_sequence_base::M_revalidate_singular () [protected],[inherited]

Revalidates all attached singular iterators. This method may be used to validate iterators that were invalidated before (but for some reason, such as an exception, need to become valid again).

5.447.3.11 void __gnu_debug::Safe_sequence_base::M_swap (_Safe_sequence_base & __x) [protected],[noexcept],[inherited]

Swap this sequence with the given sequence. This operation also swaps ownership of the iterators, so that when the operation is complete all iterators that originally referenced one container now reference the other container.

5.447.3.12 void __gnu_debug::Safe_sequence<vector<_Tp, _Allocator > >::M_transfer_from_if (_Safe_sequence<vector<_Tp, _Allocator > > & __from, _Predicate __pred) [inherited]

Transfers all iterators *x* that reference *from* sequence, are not singular, and for which __pred(*x*) returns true. __pred will be invoked with the normal iterators nested in the safe ones.

5.447.4 Member Data Documentation

5.447.4.1 `_Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_const_iterators` [inherited]

The list of constant iterators that reference this container.

Definition at line 193 of file `safe_base.h`.

Referenced by `__gnu_debug::Safe_sequence<_Sequence>::M_transfer_from_if()`.

5.447.4.2 `_Safe_iterator_base* __gnu_debug::Safe_sequence_base::M_iterators` [inherited]

The list of mutable iterators that reference this container.

Definition at line 190 of file `safe_base.h`.

Referenced by `__gnu_debug::Safe_sequence<_Sequence>::M_transfer_from_if()`.

5.447.4.3 `unsigned int __gnu_debug::Safe_sequence_base::M_version` [mutable], [inherited]

The container version number. This number may never be 0.

Definition at line 196 of file `safe_base.h`.

The documentation for this class was generated from the following file:

- [debug/vector](#)

5.448 `std::__detail::BracketMatcher<_TraitsT, __icase, __collate>` Struct Template Reference

Public Types

- `typedef _TraitsT::char_class_type _CharClassT`
- `typedef _TransT::_CharT _CharT`
- `typedef _TraitsT::string_type _StringT`
- `typedef _TransT::_StrTransT _StrTransT`
- `typedef _RegexTranslator<_TraitsT, __icase, __collate> _TransT`

Public Member Functions

- `_BracketMatcher` (bool __is_non_matching, const _TraitsT &__traits)
- `void _M_add_char` (_CharT __c)
- `void _M_add_character_class` (const _StringT &__s, bool __neg)
- `_StringT _M_add_collate_element` (const _StringT &__s)
- `void _M_add_equivalence_class` (const _StringT &__s)
- `void _M_make_range` (_CharT __l, _CharT __r)
- `void _M_ready` ()
- `bool operator()` (_CharT __ch) const

5.448.1 Detailed Description

```
template<typename _TraitsT, bool __icase, bool __collate>
struct std::__detail::_BracketMatcher<_TraitsT, __icase, __collate>
```

Matches a character range (bracket expression)

Definition at line 43 of file `regex_compiler.h`.

The documentation for this struct was generated from the following files:

- [regex_compiler.h](#)
- [regex_compiler.tcc](#)

5.449 `std::__detail::_Compiler<_TraitsT>` Class Template Reference

Public Types

- typedef `_TraitsT::char_type` `_CharT`
- typedef [regex_constants::syntax_option_type](#) `_FlagT`
- typedef const `_CharT*` `_IterT`
- typedef `_NFA<_TraitsT>` `_RegexT`

Public Member Functions

- `_Compiler` (`_IterT` __b, `_IterT` __e, const `typename _TraitsT::locale_type` &__traits, [_FlagT](#) __flags)
- [shared_ptr](#)< const `_RegexT` > `_M_get_nfa` ()

5.449.1 Detailed Description

```
template<typename _TraitsT>
class std::__detail::_Compiler<_TraitsT>
```

Builds an NFA from an input iterator range.

The `_TraitsT` type should fulfill requirements [28.3].

Definition at line 51 of file `regex_compiler.h`.

The documentation for this class was generated from the following files:

- [regex_compiler.h](#)
- [regex_compiler.tcc](#)

5.450 `std::__detail::_Default_ranged_hash` Struct Reference

5.450.1 Detailed Description

Default ranged hash function H. In principle it should be a function object composed from objects of type H1 and H2 such that $h(k, N) = h2(h1(k), N)$, but that would mean making extra copies of h1 and h2. So instead we'll just use a tag to tell class template hashtable to do that composition.

Definition at line 454 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.451 `std::__detail::_Equal_helper<_Key, _Value, _ExtractKey, _Equal, _HashCodeType, __cache_hash_code>` Struct Template Reference

5.451.1 Detailed Description

```
template<typename _Key, typename _Value, typename _ExtractKey, typename _Equal, typename _HashCodeType, bool __cache_hash_code>
struct std::__detail::_Equal_helper<_Key, _Value, _ExtractKey, _Equal, _HashCodeType, __cache_hash_code>
```

Primary class template `_Equal_helper`.

Definition at line 1310 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.452 `std::__detail::_Equal_helper<_Key, _Value, _ExtractKey, _Equal, _HashCodeType, false>` Struct Template Reference

Static Public Member Functions

- static bool **_S_equals** (const `_Equal` &__eq, const `_ExtractKey` &__extract, const `_Key` &__k, `_HashCodeType`, [_Hash_node](#)< `_Value`, false > *__n)

5.452.1 Detailed Description

```
template<typename _Key, typename _Value, typename _ExtractKey, typename _Equal, typename _HashCodeType>
struct std::__detail::_Equal_helper<_Key, _Value, _ExtractKey, _Equal, _HashCodeType, false>
```

Specialization.

Definition at line 1326 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.453 `std::__detail::_Equal_helper< _Key, _Value, _ExtractKey, _Equal, _HashCodeType, true > Struct Template Reference`

Static Public Member Functions

- static bool **_S_equals** (const _Equal &__eq, const _ExtractKey &__extract, const _Key &__k, _HashCodeType __c, [_Hash_node](#)< _Value, true > *__n)

5.453.1 Detailed Description

```
template<typename _Key, typename _Value, typename _ExtractKey, typename _Equal, typename _HashCodeType>
struct std::__detail::_Equal_helper< _Key, _Value, _ExtractKey, _Equal, _HashCodeType, true >
```

Specialization.

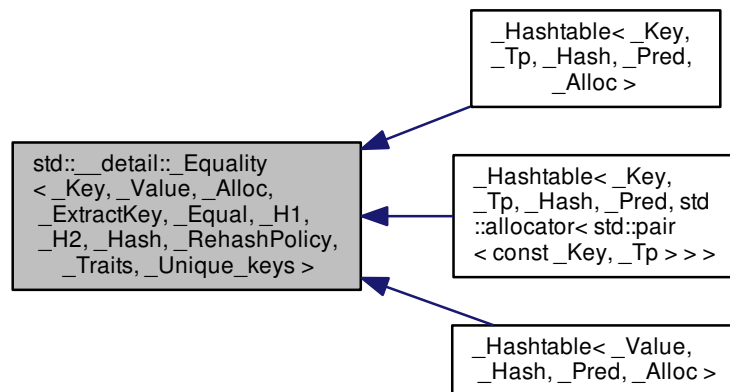
Definition at line 1315 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.454 `std::__detail::_Equality< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, _Unique_keys > Struct Template Reference`

Inheritance diagram for `std::__detail::_Equality< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, _Unique_keys >`:



5.454.1 Detailed Description

```
template<typename _Key, typename _Value, typename _Alloc, typename _ExtractKey, typename _Equal, typename _H1, typename _H2,
typename _Hash, typename _RehashPolicy, typename _Traits, bool _Unique_keys = _Traits::__unique_keys::value>
struct std::__detail::_Equality< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, _Unique_keys >
```

Primary class template `_Equality`.

This is for implementing equality comparison for unordered containers, per N3068, by John Lakos and Pablo Halpern. Algorithmically, we follow closely the reference implementations therein.

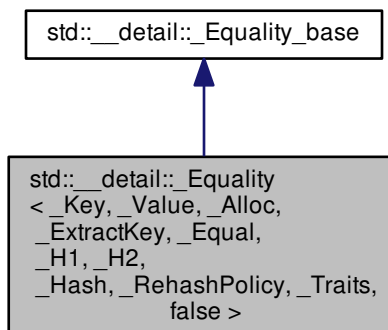
Definition at line 1791 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.455 `std::__detail::_Equality< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, false >` Struct Template Reference

Inheritance diagram for `std::__detail::_Equality< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, false >`:



Public Types

- using `__hashtable` = `_Hashtable< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits >`

Public Member Functions

- `bool _M_equal (const __hashtable &) const`

Static Protected Member Functions

- `template<typename _Uiterator >
static bool _S_is_permutation (_Uiterator, _Uiterator, _Uiterator)`

5.455.1 Detailed Description

```
template<typename _Key, typename _Value, typename _Alloc, typename _ExtractKey, typename _Equal, typename _H1, typename _H2,
typename _Hash, typename _RehashPolicy, typename _Traits>
struct std::__detail::Equality<_Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, false >
```

Specialization.

Definition at line 1836 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.456 `std::__detail::Equality<_Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, true >` Struct Template Reference

Public Types

- `using __hashtable = _Hashtable<_Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits >`

Public Member Functions

- `bool _M_equal (const __hashtable &) const`

5.456.1 Detailed Description

```
template<typename _Key, typename _Value, typename _Alloc, typename _ExtractKey, typename _Equal, typename _H1, typename _H2,
typename _Hash, typename _RehashPolicy, typename _Traits>
struct std::__detail::Equality<_Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, true >
```

Specialization.

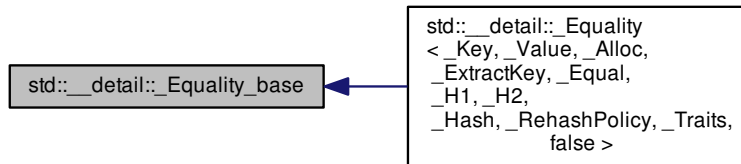
Definition at line 1798 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.457 std::__detail::_Equality_base Struct Reference

Inheritance diagram for std::__detail::_Equality_base:



Static Protected Member Functions

- `template<typename _Uiterator >`
`static bool _S_is_permutation (_Uiterator, _Uiterator, _Uiterator)`

5.457.1 Detailed Description

struct `_Equality_base`.

Common types and functions for class `_Equality`.

Definition at line 1724 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.458 std::__detail::_Executor< _Bilter, _Alloc, _TraitsT, __dfs_mode > Class Template Reference

Public Types

- `typedef iterator_traits< _Bilter >::value_type _CharT`
- `typedef _TraitsT::char_class_type _ClassT`
- `typedef regex_constants::match_flag_type _FlagT`
- `typedef _NFA< _TraitsT > _NFAT`
- `typedef basic_regex< _CharT, _TraitsT > _RegexT`
- `typedef std::vector< sub_match< _Bilter >, _Alloc > _ResultsVec`

Public Member Functions

- `_Executor` (`_Bilter __begin`, `_Bilter __end`, `_ResultsVec &__results`, `const _RegexT &__re`, `_FlagT __flags`)
- `bool _M_match ()`
- `bool _M_search ()`
- `bool _M_search_from_first ()`

Public Attributes

- `_Bilter _M_begin`
- `_ResultsVec _M_cur_results`
- `_Bilter _M_current`
- `const _Bilter _M_end`
- `_FlagT _M_flags`
- `bool _M_has_sol`
- `const _NFAT & _M_nfa`
- `const _RegexT & _M_re`
- `vector< pair< _Bilter, int > > _M_rep_count`
- `_ResultsVec & _M_results`
- `_State_info< __search_mode, _ResultsVec > _M_states`

5.458.1 Detailed Description

```
template<typename _Bilter, typename _Alloc, typename _TraitsT, bool __dfs_mode>
class std::__detail::_Executor<_Bilter, _Alloc, _TraitsT, __dfs_mode>
```

Takes a regex and an input string and does the matching.

The `_Executor` class has two modes: DFS mode and BFS mode, controlled by the template parameter `__dfs_mode`.

Definition at line 63 of file `regex.h`.

The documentation for this class was generated from the following files:

- [regex.h](#)
- [regex_executor.h](#)
- [regex_executor.tcc](#)

5.459 `std::__detail::_Hash_code_base<_Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache_hash_code >` Struct Template Reference

5.459.1 Detailed Description

```
template<typename _Key, typename _Value, typename _ExtractKey, typename _H1, typename _H2, typename _Hash, bool __cache_hash_code <←
hash_code>
struct std::__detail::_Hash_code_base<_Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache_hash_code >
```

Primary class template `_Hash_code_base`.

Encapsulates two policy issues that aren't quite orthogonal. (1) the difference between using a ranged hash function and using the combination of a hash function and a range-hashing function. In the former case we don't have such things as hash codes, so we have a dummy type as placeholder. (2) Whether or not we cache hash codes. Caching hash codes is meaningless if we have a ranged hash function.

We also put the key extraction objects here, for convenience. Each specialization derives from one or more of the template parameters to benefit from Ebo. This is important as this type is inherited in some cases by the `_Local_iterator_base` type used to implement `local_iterator` and `const_local_iterator`. As with any iterator type we prefer to make it as small as possible.

Primary template is unused except as a hook for specializations.

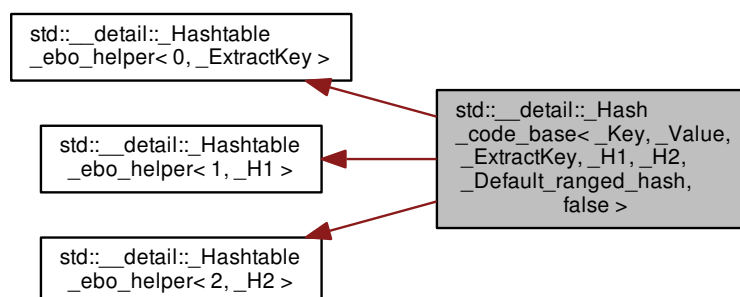
Definition at line 1047 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.460 `std::__detail::_Hash_code_base<_Key, _Value, _ExtractKey, _H1, _H2, _Default_ranged_hash, false >` Struct Template Reference

Inheritance diagram for `std::__detail::_Hash_code_base<_Key, _Value, _ExtractKey, _H1, _H2, _Default_ranged_hash, false >`:



Public Types

- typedef `_H1` **hasher**

Public Member Functions

- hasher **hash_function** () const

Protected Types

- typedef `std::size_t` **__hash_code**
- typedef `_Hash_node<_Value,false>` **__node_type**

Protected Member Functions

- **_Hash_code_base** (const `_ExtractKey` &__ex, const `_H1` &__h1, const `_H2` &__h2, const `_Default_ranged_hash` &)
- `std::size_t` **_M_bucket_index** (const `_Key` &, `__hash_code` __c, `std::size_t` __n) const
- `std::size_t` **_M_bucket_index** (const `__node_type` *__p, `std::size_t` __n) const noexcept(noexcept(declval< const `_H1` & >()(declval< const `_Key` & >()))&&noexcept(declval< const `_H2` & >()(__hash_code) 0, (std::size_t) 0)))
- void **_M_copy_code** (`__node_type` *, const `__node_type` *) const
- const `_ExtractKey` & **_M_extract** () const
- `_ExtractKey` & **_M_extract** ()
- const `_H1` & **_M_h1** () const
- `_H1` & **_M_h1** ()
- const `_H2` & **_M_h2** () const
- `_H2` & **_M_h2** ()
- `__hash_code` **_M_hash_code** (const `_Key` &__k) const
- void **_M_store_code** (`__node_type` *, `__hash_code`) const
- void **_M_swap** (`_Hash_code_base` &__x)

Friends

- struct `_Local_iterator_base<_Key,_Value,_ExtractKey,_H1,_H2,_Default_ranged_hash,false>`

5.460.1 Detailed Description

```
template<typename _Key, typename _Value, typename _ExtractKey, typename _H1, typename _H2>
struct std::__detail::__Hash_code_base<_Key,_Value,_ExtractKey,_H1,_H2,_Default_ranged_hash,false>
```

Specialization: hash function and range-hashing function, no caching of hash codes. Provides typedef and accessor required by C++ 11.

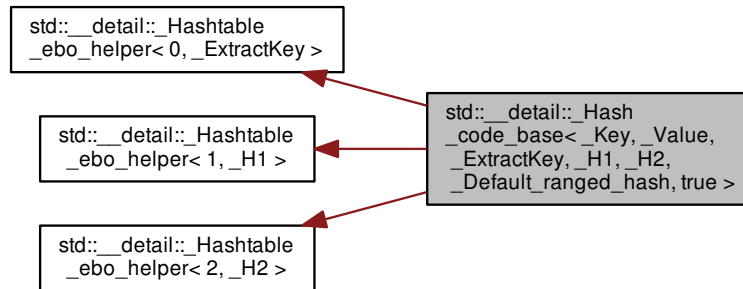
Definition at line 1130 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.461 `std::__detail::_Hash_code_base<_Key, _Value, _ExtractKey, _H1, _H2, _Default_ranged_hash, true >` Struct Template Reference

Inheritance diagram for `std::__detail::_Hash_code_base<_Key, _Value, _ExtractKey, _H1, _H2, _Default_ranged_hash, true >`:



Public Types

- typedef `_H1 hasher`

Public Member Functions

- hasher **hash_function** () const

Protected Types

- typedef `std::size_t __hash_code`
- typedef `_Hash_node<_Value, true > __node_type`

Protected Member Functions

- **_Hash_code_base** (const `_ExtractKey` &__ex, const `_H1` &__h1, const `_H2` &__h2, const `_Default_ranged_hash` &__)
- `std::size_t M_bucket_index` (const `_Key` &__, `__hash_code` __c, `std::size_t` __n) const
- `std::size_t M_bucket_index` (const `__node_type` *__p, `std::size_t` __n) const noexcept(noexcept(declval<const `_H2` & >>()((`__hash_code`) 0, (`std::size_t`) 0)))
- void **M_copy_code** (`__node_type` *__to, const `__node_type` *__from) const
- const `_ExtractKey` & **M_extract** () const
- `_ExtractKey` & **M_extract** ()
- const `_H1` & **M_h1** () const
- `_H1` & **M_h1** ()
- const `_H2` & **M_h2** () const
- `_H2` & **M_h2** ()
- `__hash_code` **M_hash_code** (const `_Key` &__k) const
- void **M_store_code** (`__node_type` *__n, `__hash_code` __c) const
- void **M_swap** (`_Hash_code_base` &__x)

Friends

- `struct _Local_iterator_base<_Key, _Value, _ExtractKey, _H1, _H2, _Default_ranged_hash, true >`

5.461.1 Detailed Description

```
template<typename _Key, typename _Value, typename _ExtractKey, typename _H1, typename _H2>
struct std::__detail::_Hash_code_base<_Key, _Value, _ExtractKey, _H1, _H2, _Default_ranged_hash, true >
```

Specialization: hash function and range-hashing function, caching hash codes. H is provided but ignored. Provides typedef and accessor required by C++ 11.

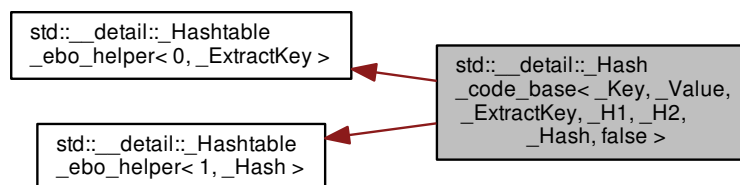
Definition at line 1220 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.462 `std::__detail::_Hash_code_base<_Key, _Value, _ExtractKey, _H1, _H2, _Hash, false >` Struct Template Reference

Inheritance diagram for `std::__detail::_Hash_code_base<_Key, _Value, _ExtractKey, _H1, _H2, _Hash, false >`:



Protected Types

- `typedef void * __hash_code`
- `typedef _Hash_node<_Value, false > __node_type`

Protected Member Functions

- `_Hash_code_base` (const `_ExtractKey` & __ex, const `_H1` &, const `_H2` &, const `_Hash` & __h)
- `std::size_t _M_bucket_index` (const `_Key` & __k, __hash_code, `std::size_t` __n) const
- `std::size_t _M_bucket_index` (const `__node_type` * __p, `std::size_t` __n) const noexcept(noexcept(declval< const `_Hash` & >()(declval< const `_Key` & >()), (std::size_t) 0)))
- `void _M_copy_code` (`__node_type` *, const `__node_type` *) const
- `const _ExtractKey & _M_extract` () const
- `_ExtractKey & _M_extract` ()
- `__hash_code _M_hash_code` (const `_Key` & __key) const
- `const _Hash & _M_ranged_hash` () const
- `_Hash & _M_ranged_hash` ()
- `void _M_store_code` (`__node_type` *, __hash_code) const
- `void _M_swap` (`_Hash_code_base` & __x)

5.462.1 Detailed Description

```
template<typename _Key, typename _Value, typename _ExtractKey, typename _H1, typename _H2, typename _Hash>
struct std::__detail::_Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, false >
```

Specialization: ranged hash function, no caching hash codes. H1 and H2 are provided but ignored. We define a dummy hash code type.

Definition at line 1053 of file hashtable_policy.h.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.463 `std::__detail::_Hash_node< _Value, _Cache_hash_code >` Struct Template Reference

5.463.1 Detailed Description

```
template<typename _Value, bool _Cache_hash_code>
struct std::__detail::_Hash_node< _Value, _Cache_hash_code >
```

Primary template struct `_Hash_node`.

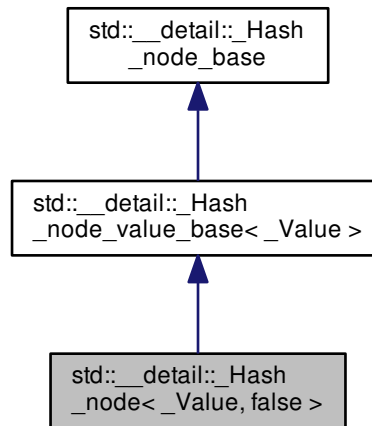
Definition at line 269 of file hashtable_policy.h.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.464 std::__detail::_Hash_node< _Value, false > Struct Template Reference

Inheritance diagram for std::__detail::_Hash_node< _Value, false >:



Public Types

- typedef _Value **value_type**

Public Member Functions

- [_Hash_node](#) * **_M_next** () const noexcept
- _Value & **_M_v** () noexcept
- const _Value & **_M_v** () const noexcept
- _Value * **_M_valptr** () noexcept
- const _Value * **_M_valptr** () const noexcept

Public Attributes

- [_Hash_node_base](#) * **_M_nxt**
- __gnu_cxx::__aligned_buffer< _Value > **_M_storage**

5.464.1 Detailed Description

```
template<typename _Value>
struct std::__detail::_Hash_node< _Value, false >
```

Specialization for nodes without caches, struct _Hash_node.

Base class is __detail::_Hash_node_value_base.

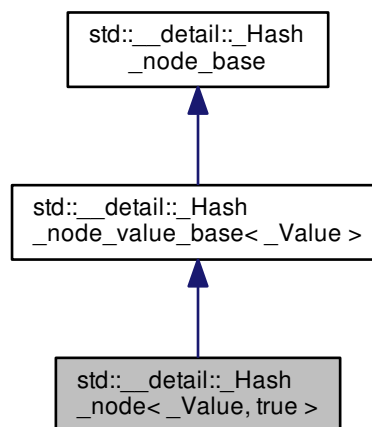
Definition at line 292 of file hashtable_policy.h.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.465 std::__detail::_Hash_node< _Value, true > Struct Template Reference

Inheritance diagram for std::__detail::_Hash_node< _Value, true >:



Public Types

- typedef _Value **value_type**

Public Member Functions

- [_Hash_node](#) * **_M_next** () const noexcept
- _Value & **_M_v** () noexcept
- const _Value & **_M_v** () const noexcept
- _Value * **_M_valptr** () noexcept
- const _Value * **_M_valptr** () const noexcept

Public Attributes

- std::size_t **M_hash_code**
- [_Hash_node_base](#) * **M_nxt**
- `__gnu_cxx::__aligned_buffer<_Value>` **M_storage**

5.465.1 Detailed Description

```
template<typename _Value>
struct std::__detail::_Hash_node<_Value, true >
```

Specialization for nodes with caches, struct `_Hash_node`.

Base class is `__detail::_Hash_node_value_base`.

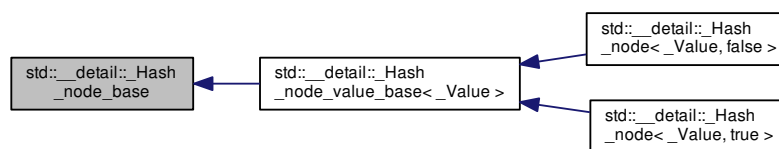
Definition at line 277 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.466 std::__detail::_Hash_node_base Struct Reference

Inheritance diagram for `std::__detail::_Hash_node_base`:



Public Member Functions

- `_Hash_node_base` (`_Hash_node_base` * __next) noexcept

Public Attributes

- [_Hash_node_base](#) * **M_nxt**

5.466.1 Detailed Description

struct `_Hash_node_base`

Nodes, used to wrap elements stored in the hash table. A policy template parameter of class template `_Hashtable` controls whether nodes also store a hash code. In some cases (e.g. strings) this may be a performance win.

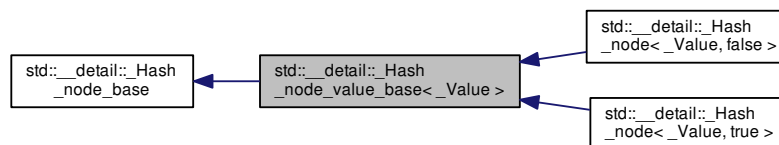
Definition at line 227 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.467 `std::__detail::_Hash_node_value_base<_Value>` Struct Template Reference

Inheritance diagram for `std::__detail::_Hash_node_value_base<_Value>`:



Public Types

- `typedef _Value value_type`

Public Member Functions

- `_Value & _M_v () noexcept`
- `const _Value & _M_v () const noexcept`
- `_Value * _M_valptr () noexcept`
- `const _Value * _M_valptr () const noexcept`

Public Attributes

- `_Hash_node_base * _M_nxt`
- `__gnu_cxx::__aligned_buffer<_Value> _M_storage`

5.467.1 Detailed Description

```
template<typename _Value>
struct std::__detail::__Hash_node_value_base< _Value >
```

```
struct _Hash_node_value_base
```

Node type with the value to store.

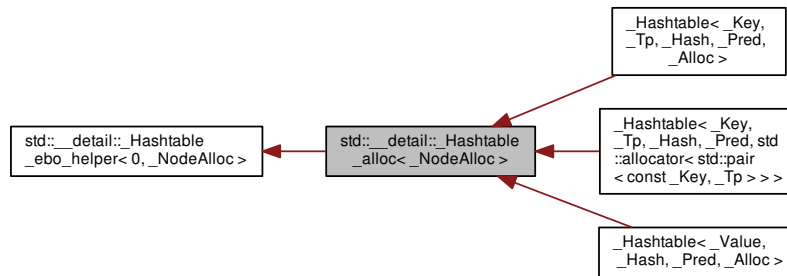
Definition at line 242 of file hashtable_policy.h.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.468 std::__detail::__Hashtable_alloc< _NodeAlloc > Struct Template Reference

Inheritance diagram for std::__detail::__Hashtable_alloc< _NodeAlloc >:



Public Types

- using `__bucket_alloc_traits` = `std::allocator_traits< __bucket_alloc_type >`
- using `__bucket_alloc_type` = `__alloc_rebind< __node_alloc_type, __bucket_type >`
- using `__bucket_type` = `__node_base *`
- using `__node_alloc_traits` = `__gnu_cxx::__alloc_traits< __node_alloc_type >`
- using `__node_alloc_type` = `_NodeAlloc`
- using `__node_base` = `__detail::__Hash_node_base`
- using `__node_type` = `typename _NodeAlloc::value_type`
- using `__value_alloc_traits` = `std::allocator_traits< __value_alloc_type >`
- using `__value_alloc_type` = `__alloc_rebind< __node_alloc_type, __value_type >`
- using `__value_type` = `typename __node_type::value_type`

Public Member Functions

- `_Hashtable_alloc` (const `_Hashtable_alloc` &)=default
- `_Hashtable_alloc` (`_Hashtable_alloc` &&)=default
- template<typename `_Alloc` >
 `_Hashtable_alloc` (`_Alloc` &&__a)
- `__bucket_type` * `_M_allocate_buckets` (std::size_t __n)
- template<typename... `_Args`>
 `__node_type` * `_M_allocate_node` (`_Args` &&...__args)
- void `_M_deallocate_buckets` (`__bucket_type` *, std::size_t __n)
- void `_M_deallocate_node` (`__node_type` * __n)
- void `_M_deallocate_nodes` (`__node_type` * __n)
- `__node_alloc_type` & `_M_node_allocator` ()
- const `__node_alloc_type` & `_M_node_allocator` () const

5.468.1 Detailed Description

```
template<typename _NodeAlloc>
struct std::__detail::_Hashtable_alloc< _NodeAlloc >
```

This type deals with all allocation and keeps an allocator instance through inheritance to benefit from EBO when possible.

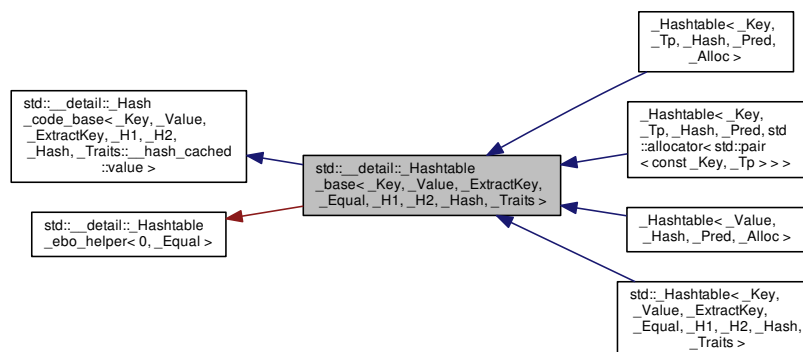
Definition at line 106 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.469 `std::__detail::_Hashtable_base< _Key, _Value, _ExtractKey, _Equal, _H1, _H2, _Hash, _Traits >` Struct Template Reference

Inheritance diagram for `std::__detail::_Hashtable_base< _Key, _Value, _ExtractKey, _Equal, _H1, _H2, _Hash, _Traits >`:



Public Types

- using `__constant_iterators` = typename `__traits_type::__constant_iterators`
- using `__hash_cached` = typename `__traits_type::__hash_cached`
- using `__hash_code` = typename `__hash_code_base::__hash_code`
- using `__hash_code_base` = `_Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __hash_↵cached::value >`
- using `__ireturn_type` = typename `std::conditional< __unique_keys::value, std::pair< iterator, bool >, iterator >::type`
- using `__node_type` = typename `__hash_code_base::__node_type`
- using `__traits_type` = `_Traits`
- using `__unique_keys` = typename `__traits_type::__unique_keys`
- using `const_iterator` = `__detail::__Node_const_iterator< value_type, __constant_iterators::value, __hash_↵cached::value >`
- using `const_local_iterator` = `__detail::__Local_const_iterator< key_type, value_type, _ExtractKey, _H1, _H2, ↵_Hash, __constant_iterators::value, __hash_cached::value >`
- typedef `std::ptrdiff_t` `difference_type`
- using `iterator` = `__detail::__Node_iterator< value_type, __constant_iterators::value, __hash_cached::value >`
- typedef `_Equal` `key_equal`
- typedef `_Key` `key_type`
- using `local_iterator` = `__detail::__Local_iterator< key_type, value_type, _ExtractKey, _H1, _H2, _Hash, ↵constant_iterators::value, __hash_cached::value >`
- typedef `std::size_t` `size_type`
- typedef `_Value` `value_type`

Protected Member Functions

- `_Hashtable_base` (const `_ExtractKey` &__ex, const `_H1` &__h1, const `_H2` &__h2, const `_Hash` &__hash, const `_Equal` &__eq)
- const `_Equal` & `_M_eq` () const
- `_Equal` & `_M_eq` ()
- bool `_M_equals` (const `_Key` &__k, `__hash_code` __c, `__node_type` *__n) const
- void `_M_swap` (`_Hashtable_base` &__x)

5.469.1 Detailed Description

```
template<typename _Key, typename _Value, typename _ExtractKey, typename _Equal, typename _H1, typename _H2, typename _Hash,
typename _Traits>
```

```
struct std::__detail::__Hashtable_base< _Key, _Value, _ExtractKey, _Equal, _H1, _H2, _Hash, _Traits >
```

Primary class template `_Hashtable_base`.

Helper class adding management of `_Equal` functor to `_Hash_code_base` type.

Base class templates are:

- `__detail::__Hash_code_base`
- `__detail::__Hashtable_ebo_helper`

Definition at line 58 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.470 `std::__detail::_Hashtable_ebo_helper<_Nm, _Tp, __use_ebo >` Struct Template Reference

5.470.1 Detailed Description

```
template<int _Nm, typename _Tp, bool __use_ebo = !__is_final(_Tp) && __is_empty(_Tp)>
struct std::__detail::_Hashtable_ebo_helper<_Nm, _Tp, __use_ebo >
```

Primary class template `_Hashtable_ebo_helper`.

Helper class using EBO when it is not forbidden (the type is not final) and when it is worth it (the type is empty.)

Definition at line 967 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.471 `std::__detail::_Hashtable_ebo_helper<_Nm, _Tp, false >` Struct Template Reference

Public Member Functions

- `template<typename _OtherTp >`
`_Hashtable_ebo_helper` (`_OtherTp &&__tp`)

Static Public Member Functions

- `static const _Tp & _S_cget` (`const _Hashtable_ebo_helper &__eboh`)
- `static _Tp & _S_get` (`_Hashtable_ebo_helper &__eboh`)

5.471.1 Detailed Description

```
template<int _Nm, typename _Tp>
struct std::__detail::_Hashtable_ebo_helper<_Nm, _Tp, false >
```

Specialization not using EBO.

Definition at line 992 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.472 `std::__detail::_Hashtable_ebo_helper<_Nm, _Tp, true >` Struct Template Reference

Inherits `_Tp`.

Public Member Functions

- `template<typename _OtherTp >`
`__Hashtable_ebo_helper` (`_OtherTp` && `__tp`)

Static Public Member Functions

- `static const _Tp & __S_cget` (`const __Hashtable_ebo_helper` & `__eboh`)
- `static _Tp & __S_get` (`__Hashtable_ebo_helper` & `__eboh`)

5.472.1 Detailed Description

```
template<int _Nm, typename _Tp>
struct std::__detail::_Hashtable_ebo_helper< _Nm, _Tp, true >
```

Specialization using EBO.

Definition at line 971 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.473 `std::__detail::_Hashtable_traits< _Cache_hash_code, _Constant_iterators, _Unique_keys >` Struct Template Reference

Public Types

- using `__constant_iterators` = `__bool_constant< _Constant_iterators >`
- using `__hash_cached` = `__bool_constant< _Cache_hash_code >`
- using `__unique_keys` = `__bool_constant< _Unique_keys >`

5.473.1 Detailed Description

```
template<bool _Cache_hash_code, bool _Constant_iterators, bool _Unique_keys>
struct std::__detail::_Hashtable_traits< _Cache_hash_code, _Constant_iterators, _Unique_keys >
```

`struct _Hashtable_traits`

Important traits for hash tables.

Template Parameters

<code>_Cache_hash_code</code>	Boolean value. True if the value of the hash function is stored along with the value. This is a time-space tradeoff. Storing it may improve lookup speed by reducing the number of times we need to call the <code>_Equal</code> function.
<code>_Constant_iterators</code>	Boolean value. True if iterator and <code>const_iterator</code> are both constant iterator types. This is true for <code>unordered_set</code> and <code>unordered_multiset</code> , false for <code>unordered_map</code> and <code>unordered_multimap</code> .
<code>_Unique_keys</code>	Boolean value. True if the return value of <code>_Hashtable::count(k)</code> is always at most one, false if it may be an arbitrary number. This is true for <code>unordered_set</code> and <code>unordered_map</code> , false for <code>unordered_multiset</code> and <code>unordered_multimap</code> .

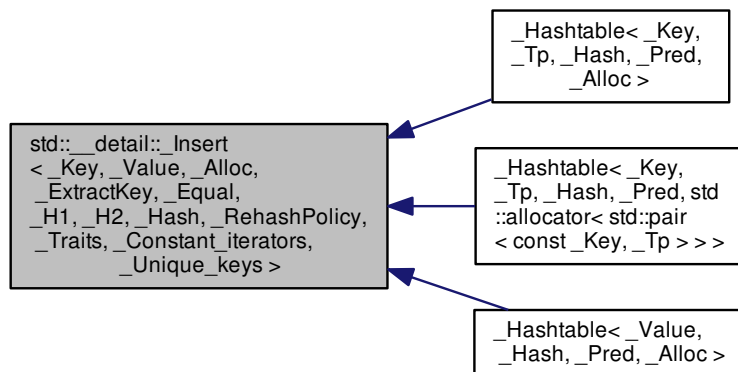
Definition at line 212 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.474 `std::__detail::Insert<_Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, _Constant_iterators, _Unique_keys>` Struct Template Reference

Inheritance diagram for `std::__detail::Insert<_Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, _Constant_iterators, _Unique_keys>`:



5.474.1 Detailed Description

```
template<typename _Key, typename _Value, typename _Alloc, typename _ExtractKey, typename _Equal, typename _H1, typename _H2, typename _Hash, typename _RehashPolicy, typename _Traits, bool _Constant_iterators = _Traits::__constant_iterators::value, bool _Unique_keys = _Traits::__unique_keys::value>
```

```
struct std::detail::Insert<_Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, _Constant_iterators, _Unique_keys>
```

Primary class template `_Insert`.

Select insert member functions appropriate to `_Hashtable` policy choices.

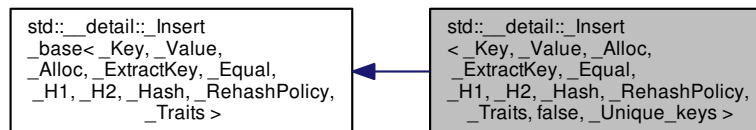
Definition at line 780 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.475 `std::detail::Insert<_Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, false, _Unique_keys>` Struct Template Reference

Inheritance diagram for `std::detail::Insert<_Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, false, _Unique_keys>`:



Public Types

- using `__base_type` = `_Insert_base<_Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits>`
- using `__hashtable` = `typename __base_type::__hashtable`
- using `__ireturn_type` = `typename __base_type::__ireturn_type`
- `template<typename _Pair>`
using `__is_cons` = `std::is_constructible<value_type, _Pair &&>`
- using `__unique_keys` = `typename __base_type::__unique_keys`
- `template<typename _Pair>`
using `__IFcons` = `std::enable_if<__is_cons<_Pair>::value>`
- `template<typename _Pair>`
using `__IFconsp` = `typename __IFcons<_Pair>::type`
- using `const_iterator` = `typename __base_type::const_iterator`
- using `iterator` = `typename __base_type::iterator`
- using `value_type` = `typename __base_type::value_type`

Public Member Functions

- `__ireturn_type insert` (`const value_type &__v`)
- `iterator insert` (`const_iterator __hint, const value_type &__v`)
- `void insert` (`initializer_list< value_type > __l`)
- `template<typename _InputIterator >`
`void insert` (`_InputIterator __first, _InputIterator __last`)
- `template<typename _Pair, typename = _IFconsp<_Pair>>`
`__ireturn_type insert` (`_Pair &&__v`)
- `template<typename _Pair, typename = _IFconsp<_Pair>>`
`iterator insert` (`const_iterator __hint, _Pair &&__v`)

Protected Types

- `using __hashtable_base = _Hashtable_base< _Key, _Value, _ExtractKey, _Equal, _H1, _H2, _Hash, _Traits >`
- `using __node_alloc_type = __alloc_rebind< _Alloc, __node_type >`
- `using __node_gen_type = _AllocNode< __node_alloc_type >`
- `using __node_type = _Hash_node< _Value, _Traits::__hash_cached::value >`
- `using size_type = typename __hashtable_base::size_type`

Protected Member Functions

- `__hashtable & _M_conjure_hashtable` ()
- `template<typename _InputIterator, typename _NodeGetter >`
`void _M_insert_range` (`_InputIterator __first, _InputIterator __last, const _NodeGetter &`)

5.475.1 Detailed Description

```
template<typename _Key, typename _Value, typename _Alloc, typename _ExtractKey, typename _Equal, typename _H1, typename _H2,
typename _Hash, typename _RehashPolicy, typename _Traits, bool _Unique_keys>
struct std::__detail::Insert< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, false, _Unique_keys
>
```

Specialization.

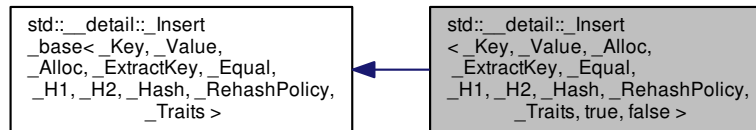
Definition at line 869 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.476 `std::__detail::Insert<_Key,_Value,_Alloc,_ExtractKey,_Equal,_H1,_H2,_Hash,_RehashPolicy,_Traits,true,false>` Struct Template Reference

Inheritance diagram for `std::__detail::Insert<_Key,_Value,_Alloc,_ExtractKey,_Equal,_H1,_H2,_Hash,_RehashPolicy,_Traits,true,false>`:



Public Types

- using `__base_type` = `Insert_base<_Key,_Value,_Alloc,_ExtractKey,_Equal,_H1,_H2,_Hash,_RehashPolicy,_Traits>`
- using `__hashtable` = `typename __base_type::__hashtable`
- using `__node_gen_type` = `typename __base_type::__node_gen_type`
- using `__unique_keys` = `typename __base_type::__unique_keys`
- using `const_iterator` = `typename __base_type::const_iterator`
- using `iterator` = `typename __base_type::iterator`
- using `value_type` = `typename __base_type::value_type`

Public Member Functions

- `__ireturn_type insert` (`const value_type &__v`)
- `iterator insert` (`const_iterator __hint, const value_type &__v`)
- `void insert` (`initializer_list<value_type> __l`)
- `template<typename _InputIterator> void insert` (`_InputIterator __first, _InputIterator __last`)
- `iterator insert` (`value_type &&__v`)
- `iterator insert` (`const_iterator __hint, value_type &&__v`)

Protected Types

- using `__hashtable_base` = `Hashtable_base<_Key,_Value,_ExtractKey,_Equal,_H1,_H2,_Hash,_Traits>`
- using `__ireturn_type` = `typename __hashtable_base::__ireturn_type`
- using `__node_alloc_type` = `_alloc_rebind<_Alloc, __node_type>`
- using `__node_type` = `Hash_node<_Value,_Traits::__hash_cached::value>`
- using `size_type` = `typename __hashtable_base::size_type`

Protected Member Functions

- [__hashtable](#) & [_M_conjure_hashtable](#) ()
- `template<typename _InputIterator, typename _NodeGetter >`
`void _M_insert_range (_InputIterator __first, _InputIterator __last, const _NodeGetter &)`

5.476.1 Detailed Description

```
template<typename _Key, typename _Value, typename _Alloc, typename _ExtractKey, typename _Equal, typename _H1, typename _H2,
typename _Hash, typename _RehashPolicy, typename _Traits>
struct std::__detail::Insert< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, true, false >
```

Specialization.

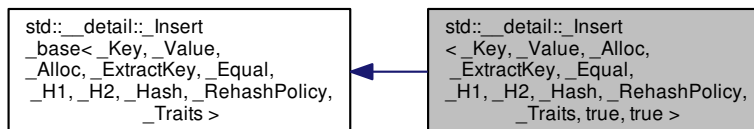
Definition at line 828 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.477 `std::__detail::Insert< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, true, true >` Struct Template Reference

Inheritance diagram for `std::__detail::Insert< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, true, true >`:



Public Types

- using `__base_type` = `_Insert_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits >`
- using `__hashtable` = `typename __base_type::__hashtable`
- using `__node_gen_type` = `typename __base_type::__node_gen_type`
- using `__unique_keys` = `typename __base_type::__unique_keys`
- using `const_iterator` = `typename __base_type::const_iterator`
- using `iterator` = `typename __base_type::iterator`
- using `value_type` = `typename __base_type::value_type`

Public Member Functions

- `__ireturn_type insert` (`const value_type &__v`)
- iterator `insert` (`const_iterator __hint, const value_type &__v`)
- void `insert` (`initializer_list<value_type> __l`)
- `template<typename _InputIterator>`
void `insert` (`_InputIterator __first, _InputIterator __last`)
- `std::pair<iterator, bool>` `insert` (`value_type &&__v`)
- iterator `insert` (`const_iterator __hint, value_type &&__v`)

Protected Types

- using `__hashtable_base` = `_Hashtable_base<_Key, _Value, _ExtractKey, _Equal, _H1, _H2, _Hash, _Traits>`
- using `__ireturn_type` = `typename __hashtable_base::__ireturn_type`
- using `__node_alloc_type` = `_alloc_rebind<_Alloc, __node_type>`
- using `__node_type` = `_Hash_node<_Value, _Traits::__hash_cached::value>`
- using `size_type` = `typename __hashtable_base::size_type`

Protected Member Functions

- `__hashtable & _M_conjure_hashtable` ()
- `template<typename _InputIterator, typename _NodeGetter>`
void `_M_insert_range` (`_InputIterator __first, _InputIterator __last, const _NodeGetter &`)

5.477.1 Detailed Description

```
template<typename _Key, typename _Value, typename _Alloc, typename _ExtractKey, typename _Equal, typename _H1, typename _H2,
typename _Hash, typename _RehashPolicy, typename _Traits>
struct std::__detail::Insert<_Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, true, true >
```

Specialization.

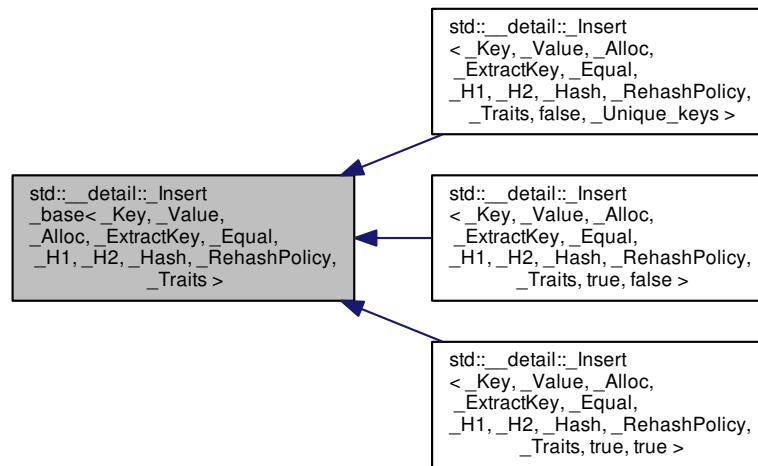
Definition at line 787 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.478 `std::__detail::_Insert_base<_Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits >` Struct Template Reference

Inheritance diagram for `std::__detail::_Insert_base<_Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits >`:



Public Member Functions

- `__ireturn_type insert` (const value_type &__v)
- iterator `insert` (const_iterator __hint, const value_type &__v)
- void `insert` (initializer_list<value_type> __l)
- template<typename _InputIterator >
void `insert` (_InputIterator __first, _InputIterator __last)

Protected Types

- using `__hashtable` = `_Hashtable`<_Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits >
- using `__hashtable_base` = `_Hashtable_base`<_Key, _Value, _ExtractKey, _Equal, _H1, _H2, _Hash, _Traits >
- using `__ireturn_type` = typename `__hashtable_base::__ireturn_type`
- using `__node_alloc_type` = `_alloc_rebind`<_Alloc, `__node_type` >
- using `__node_gen_type` = `_AllocNode`<__node_alloc_type >
- using `__node_type` = `_Hash_node`<_Value, Traits::__hash_cached::value >
- using `__unique_keys` = typename `__hashtable_base::__unique_keys`
- using `const_iterator` = typename `__hashtable_base::const_iterator`
- using `iterator` = typename `__hashtable_base::iterator`
- using `size_type` = typename `__hashtable_base::size_type`
- using `value_type` = typename `__hashtable_base::value_type`

Protected Member Functions

- [__hashtable](#) & [_M_conjure_hashtable](#) ()
- `template<typename _InputIterator, typename _NodeGetter >`
`void _M_insert_range (_InputIterator __first, _InputIterator __last, const _NodeGetter &)`

5.478.1 Detailed Description

```
template<typename _Key, typename _Value, typename _Alloc, typename _ExtractKey, typename _Equal, typename _H1, typename _H2,
typename _Hash, typename _RehashPolicy, typename _Traits>
struct std::__detail::_Insert_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits >
```

Primary class template `_Insert_base`.

insert member functions appropriate to all `_Hashtables`.

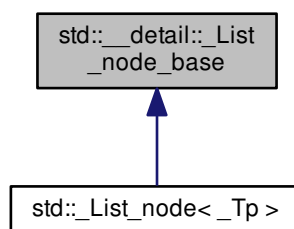
Definition at line 676 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.479 std::__detail::_List_node_base Struct Reference

Inheritance diagram for `std::__detail::_List_node_base`:



Public Member Functions

- `void _M_hook (_List_node_base *const __position) noexcept`
- `void _M_reverse () noexcept`
- `void _M_transfer (_List_node_base *const __first, _List_node_base *const __last) noexcept`
- `void _M_unhook () noexcept`

Static Public Member Functions

- static void **swap** ([_List_node_base](#) &__x, [_List_node_base](#) &__y) noexcept

Public Attributes

- [_List_node_base](#) * **_M_next**
- [_List_node_base](#) * **_M_prev**

5.479.1 Detailed Description

Common part of a node in the list.

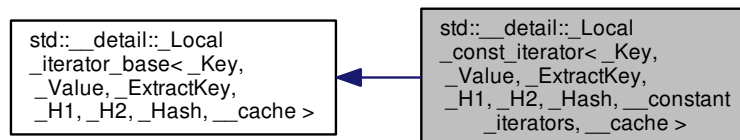
Definition at line 80 of file [stl_list.h](#).

The documentation for this struct was generated from the following file:

- [stl_list.h](#)

5.480 `std::__detail::__Local_const_iterator< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __constant_iterators, __cache >` Struct Template Reference

Inheritance diagram for `std::__detail::__Local_const_iterator< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __constant_iterators, __cache >`:



Public Types

- typedef `std::ptrdiff_t` **difference_type**
- typedef [std::forward_iterator_tag](#) **iterator_category**
- typedef `const _Value *` **pointer**
- typedef `const _Value &` **reference**
- typedef `_Value` **value_type**

Public Member Functions

- **_Local_const_iterator** (const __hash_code_base &__base, [_Hash_node](#)< _Value, __cache > *__p, std::size_t __t __bkt, std::size_t __bkt_count)
- **_Local_const_iterator** (const [_Local_iterator](#)< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __constant_iterators, __cache > &__x)
- reference **operator*** () const
- [_Local_const_iterator](#) & **operator++** ()
- [_Local_const_iterator](#) **operator++** (int)
- pointer **operator->** () const

5.480.1 Detailed Description

```
template<typename _Key, typename _Value, typename _ExtractKey, typename _H1, typename _H2, typename _Hash, bool __constant_iterators, bool __cache>
struct std::__detail::Local_const_iterator< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __constant_iterators, __cache >
```

local const_iterators

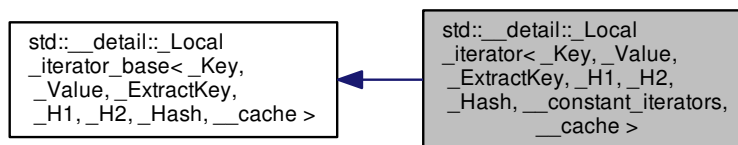
Definition at line 1572 of file hashtable_policy.h.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.481 std::__detail:: Local_iterator< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __constant_iterators, __cache > Struct Template Reference

Inheritance diagram for std::__detail:: Local_iterator< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __constant_iterators, __cache >:



Public Types

- typedef std::ptrdiff_t **difference_type**
- typedef [std::forward_iterator_tag](#) **iterator_category**
- typedef std::conditional< __constant_iterators, const _Value *, _Value * >::type **pointer**
- typedef std::conditional< __constant_iterators, const _Value &, _Value & >::type **reference**
- typedef _Value **value_type**

Public Member Functions

- **_Local_iterator** (const __hash_code_base &__base, [_Hash_node](#)< _Value, __cache > *__p, std::size_t __bkt, std::size_t __bkt_count)
- reference **operator*** () const
- [_Local_iterator](#) & **operator++** ()
- [_Local_iterator](#) **operator++** (int)
- pointer **operator->** () const

5.481.1 Detailed Description

```
template<typename _Key, typename _Value, typename _ExtractKey, typename _H1, typename _H2, typename _Hash, bool __↵
constant_iterators, bool __cache>
struct std::__detail::_Local_iterator< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __constant_iterators, __cache >
```

local iterators

Definition at line 1517 of file hashtable_policy.h.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.482 std::__detail::_Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache_hash_code > Struct Template Reference

5.482.1 Detailed Description

```
template<typename _Key, typename _Value, typename _ExtractKey, typename _H1, typename _H2, typename _Hash, bool __cache_↵
hash_code>
struct std::__detail::_Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache_hash_code >
```

Primary class template `_Local_iterator_base`.

Base class for local iterators, used to iterate within a bucket but not between buckets.

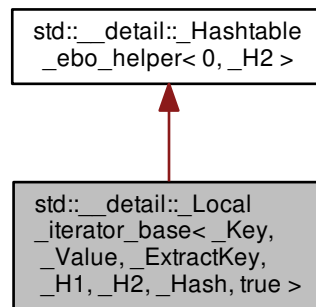
Definition at line 1022 of file hashtable_policy.h.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.483 `std::__detail::__Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, true > Struct` Template Reference

Inheritance diagram for `std::__detail::__Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, true >`:



Public Member Functions

- `const void * _M_curr () const`
- `std::size_t _M_get_bucket () const`

Protected Types

- using `__base_type` = `_Hashtable_ebo_helper< 0, _H2 >`
- using `__hash_code_base` = `_Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, true >`

Protected Member Functions

- `_Local_iterator_base` (`const __hash_code_base &__base, _Hash_node< _Value, true > *__p, std::size_t __bkt, std::size_t __bkt_count`)
- `void _M_incr ()`

Protected Attributes

- `std::size_t _M_bucket`
- `std::size_t _M_bucket_count`
- `_Hash_node< _Value, true > * _M_cur`

5.483.1 Detailed Description

```
template<typename _Key, typename _Value, typename _ExtractKey, typename _H1, typename _H2, typename _Hash>
struct std::__detail::__Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, true >
```

Partial specialization used when nodes contain a cached hash code.

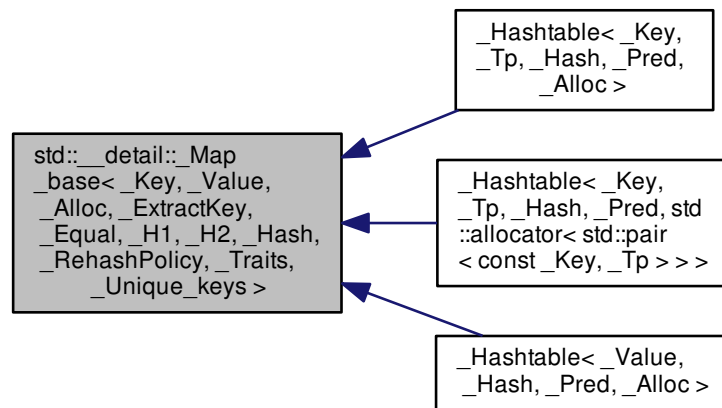
Definition at line 1338 of file hashtable_policy.h.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.484 std::__detail::__Map_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, _Unique_keys > Struct Template Reference

Inheritance diagram for std::__detail::__Map_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, _Unique_keys >:



5.484.1 Detailed Description

```
template<typename _Key, typename _Value, typename _Alloc, typename _ExtractKey, typename _Equal, typename _H1, typename _H2,
typename _Hash, typename _RehashPolicy, typename _Traits, bool _Unique_keys = _Traits::__unique_keys::value>
struct std::__detail::__Map_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, _Unique_keys >
```

Primary class template `_Map_base`.

If the hashtable has a value type of the form `pair<T1, T2>` and a key extraction policy (`_ExtractKey`) that returns the first part of the pair, the hashtable gets a `mapped_type` typedef. If it satisfies those criteria and also has unique keys, then it also gets an `operator[]`.

Definition at line 527 of file hashtable_policy.h.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.485 `std::__detail::_Map_base<_Key, _Pair, _Alloc, _Select1st, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, false >` Struct Template Reference

Public Types

- using **mapped_type** = typename [std::tuple_element](#)< 1, _Pair >::type

5.485.1 Detailed Description

```
template<typename _Key, typename _Pair, typename _Alloc, typename _Equal, typename _H1, typename _H2, typename _Hash, type-
name _RehashPolicy, typename _Traits>
struct std::__detail::_Map_base<_Key, _Pair, _Alloc, _Select1st, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, false >
```

Partial specialization, `__unique_keys` set to false.

Definition at line 533 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.486 `std::__detail::_Map_base<_Key, _Pair, _Alloc, _Select1st, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, true >` Struct Template Reference

Public Types

- using **iterator** = typename [__hashtable_base::iterator](#)
- using **key_type** = typename [__hashtable_base::key_type](#)
- using **mapped_type** = typename [std::tuple_element](#)< 1, _Pair >::type

Public Member Functions

- `mapped_type & at (const key_type &__k)`
- `const mapped_type & at (const key_type &__k) const`
- `mapped_type & operator[] (const key_type &__k)`
- `mapped_type & operator[] (key_type &&__k)`

5.486.1 Detailed Description

```
template<typename _Key, typename _Pair, typename _Alloc, typename _Equal, typename _H1, typename _H2, typename _Hash, type-
name _RehashPolicy, typename _Traits>
struct std::__detail::_Map_base<_Key, _Pair, _Alloc, _Select1st, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, true >
```

Partial specialization, `__unique_keys` set to true.

Definition at line 543 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.487 `std::__detail::_Mod_range_hashing` Struct Reference

Public Types

- typedef `std::size_t` **first_argument_type**
- typedef `std::size_t` **result_type**
- typedef `std::size_t` **second_argument_type**

Public Member Functions

- `result_type` **operator()** (`first_argument_type` __num, `second_argument_type` __den) const noexcept

5.487.1 Detailed Description

Default range hashing function: use division to fold a large number into the range [0, N).

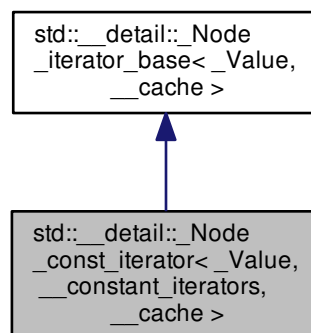
Definition at line 437 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.488 `std::__detail::_Node_const_iterator< _Value, __constant_iterators, __cache >` Struct Template Reference

Inheritance diagram for `std::__detail::_Node_const_iterator< _Value, __constant_iterators, __cache >`:



Public Types

- typedef `std::ptrdiff_t` **difference_type**
- typedef [std::forward_iterator_tag](#) **iterator_category**
- typedef `const _Value *` **pointer**
- typedef `const _Value &` **reference**
- typedef `_Value` **value_type**

Public Member Functions

- **_Node_const_iterator** (`__node_type *__p`) `noexcept`
- **_Node_const_iterator** (`const _Node_iterator<_Value, __constant_iterators, __cache > &__x`) `noexcept`
- `void` **_M_incr** () `noexcept`
- `reference` **operator*** () `const noexcept`
- **_Node_const_iterator** & **operator++** () `noexcept`
- **_Node_const_iterator** **operator++** (`int`) `noexcept`
- `pointer` **operator->** () `const noexcept`

Public Attributes

- `__node_type *` **_M_cur**

5.488.1 Detailed Description

```
template<typename _Value, bool __constant_iterators, bool __cache>
struct std::__detail::_Node_const_iterator<_Value, __constant_iterators, __cache >
```

Node `const_iterators`, used to iterate through all the hashtable.

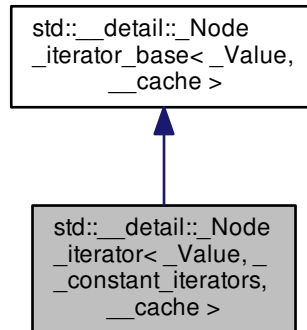
Definition at line 382 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.489 `std::__detail::_Node_iterator< _Value, __constant_iterators, __cache >` Struct Template Reference

Inheritance diagram for `std::__detail::_Node_iterator< _Value, __constant_iterators, __cache >`:



Public Types

- typedef `std::ptrdiff_t` **difference_type**
- typedef `std::forward_iterator_tag` **iterator_category**
- using **pointer** = typename `std::conditional< __constant_iterators, const _Value *, _Value * >::type`
- using **reference** = typename `std::conditional< __constant_iterators, const _Value &, _Value & >::type`
- typedef `_Value` **value_type**

Public Member Functions

- **_Node_iterator** (`__node_type * __p`) noexcept
- void **_M_incr** () noexcept
- reference **operator*** () const noexcept
- **_Node_iterator** & **operator++** () noexcept
- **_Node_iterator** **operator++** (int) noexcept
- pointer **operator->** () const noexcept

Public Attributes

- `__node_type * _M_cur`

5.489.1 Detailed Description

```
template<typename _Value, bool __constant_iterators, bool __cache>
struct std::__detail::_Node_iterator<_Value, __constant_iterators, __cache>
```

Node iterators, used to iterate through all the hashtable.

Definition at line 331 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.490 `std::__detail::_Node_iterator_base<_Value, _Cache_hash_code>` Struct Template Reference

Public Types

- using `__node_type` = [_Hash_node](#)<_Value, _Cache_hash_code>

Public Member Functions

- `_Node_iterator_base` ([__node_type](#) *__p) noexcept
- `void _M_incr` () noexcept

Public Attributes

- [__node_type](#) * `_M_cur`

5.490.1 Detailed Description

```
template<typename _Value, bool _Cache_hash_code>
struct std::__detail::_Node_iterator_base<_Value, _Cache_hash_code>
```

Base class for node iterators.

Definition at line 301 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.491 `std::__detail::_Prime_rehash_policy` Struct Reference

Public Types

- `typedef std::size_t _State`

Public Member Functions

- `_Prime_rehash_policy` (float __z=1.0) noexcept
- `std::size_t _M_bkt_for_elements` (std::size_t __n) const
- `std::pair< bool, std::size_t > _M_need_rehash` (std::size_t __n_bkt, std::size_t __n_elt, std::size_t __n_ins) const
- `std::size_t _M_next_bkt` (std::size_t __n) const
- `void _M_reset` () noexcept
- `void _M_reset` (_State __state)
- `_State _M_state` () const
- `float max_load_factor` () const noexcept

Public Attributes

- `float _M_max_load_factor`
- `std::size_t _M_next_resize`

Static Public Attributes

- `static const std::size_t _S_growth_factor`

5.491.1 Detailed Description

Default value for rehash policy. Bucket size is (usually) the smallest prime that keeps the load factor small enough.

Definition at line 458 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.492 `std::__detail::_Quoted_string< _String, _CharT >` Struct Template Reference

Public Member Functions

- `_Quoted_string` (_String __str, _CharT __del, _CharT __esc)
- `_Quoted_string & operator=` (_Quoted_string &)=delete

Public Attributes

- `_CharT _M_delim`
- `_CharT _M_escape`
- `_String _M_string`

5.492.1 Detailed Description

```
template<typename _String, typename _CharT>
struct std::__detail::__Quoted_string< _String, _CharT >
```

Struct for delimited strings.

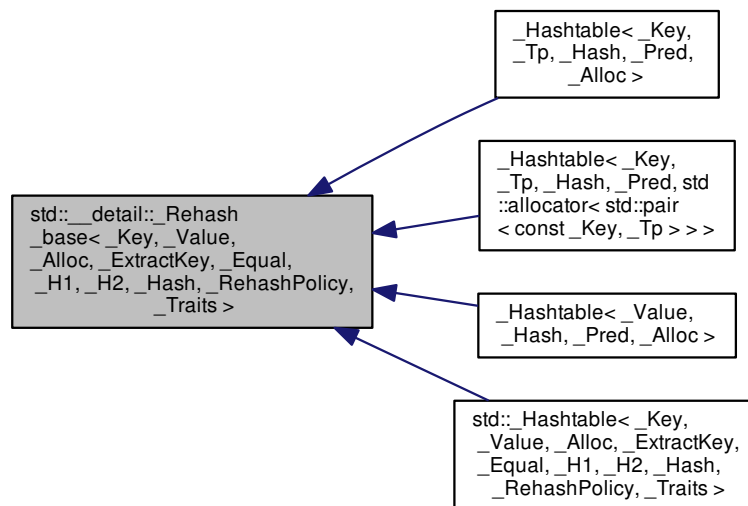
Definition at line 49 of file `quoted_string.h`.

The documentation for this struct was generated from the following file:

- [quoted_string.h](#)

5.493 `std::__detail::__Rehash_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits >` Struct Template Reference

Inheritance diagram for `std::__detail::__Rehash_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits >`:



5.493.1 Detailed Description

```
template<typename _Key, typename _Value, typename _Alloc, typename _ExtractKey, typename _Equal, typename _H1, typename _H2,
typename _Hash, typename _RehashPolicy, typename _Traits>
struct std::__detail::_Rehash_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits >
```

Primary class template `_Rehash_base`.

Give hashtable the `max_load_factor` functions and `reserve` iff the rehash policy is `_Prime_rehash_policy`.

Definition at line 924 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.494 `std::__detail::_Rehash_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _Prime_rehash_policy, _Traits >` Struct Template Reference

Public Types

- using `__hashtable` = `_Hashtable< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _Prime_rehash_policy, _Traits >`

Public Member Functions

- float **`max_load_factor`** () const noexcept
- void **`max_load_factor`** (float __z)
- void **`reserve`** (std::size_t __n)

5.494.1 Detailed Description

```
template<typename _Key, typename _Value, typename _Alloc, typename _ExtractKey, typename _Equal, typename _H1, typename _H2,
typename _Hash, typename _Traits>
struct std::__detail::_Rehash_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _Prime_rehash_policy, _Traits >
```

Specialization.

Definition at line 930 of file `hashtable_policy.h`.

The documentation for this struct was generated from the following file:

- [hashtable_policy.h](#)

5.495 `std::__detail::_Scanner<_CharT>` Class Template Reference

Inherits `std::__detail::_ScannerBase`.

Public Types

- typedef const [std::ctype](#)<_CharT> **_CtypeT**
- typedef [regex_constants::syntax_option_type](#) **_FlagT**
- typedef const _CharT * **_IterT**
- typedef [std::basic_string](#)<_CharT> **_StringT**
- enum **_TokenT** {
 _S_token_anychar, **_S_token_ord_char**, **_S_token_oct_num**, **_S_token_hex_num**,
 _S_token_backref, **_S_token_subexpr_begin**, **_S_token_subexpr_no_group_begin**, **_S_token_subexpr_end**,
 _S_token_lookahead_begin,
 _S_token_subexpr_end, **_S_token_bracket_begin**, **_S_token_bracket_neg_begin**, **_S_token_bracket_end**,
 _S_token_interval_begin, **_S_token_interval_end**, **_S_token_quoted_class**, **_S_token_char_class_name**,
 _S_token_collsymbol, **_S_token_equiv_class_name**, **_S_token_opt**, **_S_token_or**,
 _S_token_closure0, **_S_token_closure1**, **_S_token_line_begin**, **_S_token_line_end**,
 _S_token_word_bound, **_S_token_comma**, **_S_token_dup_count**, **_S_token_eof**,
 _S_token_unknown }

Public Member Functions

- **_Scanner** (_IterT __begin, _IterT __end, [_FlagT](#) __flags, [std::locale](#) __loc)
- void **_M_advance** ()
- [_TokenT](#) **_M_get_token** () const
- const [_StringT](#) & **_M_get_value** () const

Protected Types

- enum **_StateT** { **_S_state_normal**, **_S_state_in_brace**, **_S_state_in_bracket** }

Protected Member Functions

- const char * **_M_find_escape** (char __c)
- bool **_M_is_awk** () const
- bool **_M_is_basic** () const
- bool **_M_is_ecma** () const
- bool **_M_is_extended** () const
- bool **_M_is_grep** () const

Protected Attributes

- `bool _M_at_bracket_start`
- `const std::pair< char, char > _M_awk_escape_tbl [11]`
- `const char * _M_basic_spec_char`
- `const std::pair< char, char > _M_ecma_escape_tbl [8]`
- `const char * _M_ecma_spec_char`
- `const std::pair< char, char > * _M_escape_tbl`
- `const char * _M_extended_spec_char`
- `_FlagT _M_flags`
- `const char * _M_spec_char`
- `_StateT _M_state`
- `_TokenT _M_token`
- `const std::pair< char, _TokenT > _M_token_tbl [9]`

5.495.1 Detailed Description

```
template<typename _CharT>
class std::__detail::_Scanner< _CharT >
```

Scans an input range for regex tokens.

The `_Scanner` class interprets the regular expression pattern in the input range passed to its constructor as a sequence of parse tokens passed to the regular expression compiler. The sequence of tokens provided depends on the flag settings passed to the constructor: different regular expression grammars will interpret the same input pattern in syntactically different ways.

Definition at line 209 of file `regex_scanner.h`.

5.495.2 Member Enumeration Documentation

5.495.2.1 `enum std::__detail::_ScannerBase::_TokenT` `[inherited]`

Token types returned from the scanner.

Definition at line 46 of file `regex_scanner.h`.

The documentation for this class was generated from the following files:

- [regex_scanner.h](#)
- [regex_scanner.tcc](#)

5.496 `std::__detail::_StateSeq< _TraitsT >` Class Template Reference

Public Types

- `typedef _NFA< _TraitsT > _RegexT`

Public Member Functions

- **_StateSeq** (_RegexT &__nfa, _StateIdT __s)
- **_StateSeq** (_RegexT &__nfa, _StateIdT __s, _StateIdT __end)
- void **_M_append** (_StateIdT __id)
- void **_M_append** (const **_StateSeq** &__s)
- **_StateSeq** **_M_clone** ()

Public Attributes

- **_StateIdT** **_M_end**
- **_RegexT** & **_M_nfa**
- **_StateIdT** **_M_start**

5.496.1 Detailed Description

```
template<typename _TraitsT>
class std::__detail::_StateSeq<_TraitsT >
```

Describes a sequence of one or more **_State**, its current start and end(s). This structure contains fragments of an NFA during construction.

Definition at line 354 of file `regex_automaton.h`.

The documentation for this class was generated from the following files:

- [regex_automaton.h](#)
- [regex_automaton.tcc](#)

5.497 std::__exception_ptr::exception_ptr Class Reference

Public Member Functions

- **exception_ptr** (const **exception_ptr** &) noexcept
- **exception_ptr** (nullptr_t) noexcept
- **exception_ptr** (**exception_ptr** &&__o) noexcept
- const class **std::type_info** * **__cxa_exception_type** () const noexcept __attribute__((__pure__))
- **operator bool** () const
- **exception_ptr** & **operator=** (const **exception_ptr** &) noexcept
- **exception_ptr** & **operator=** (**exception_ptr** &&__o) noexcept
- void **swap** (**exception_ptr** &) noexcept

Friends

- bool **operator==** (const **exception_ptr** &, const **exception_ptr** &) noexcept __attribute__((__pure__))
- **exception_ptr** **std::current_exception** () noexcept
- void **std::rethrow_exception** (**exception_ptr**)

5.497.1 Detailed Description

An opaque pointer to an arbitrary exception.

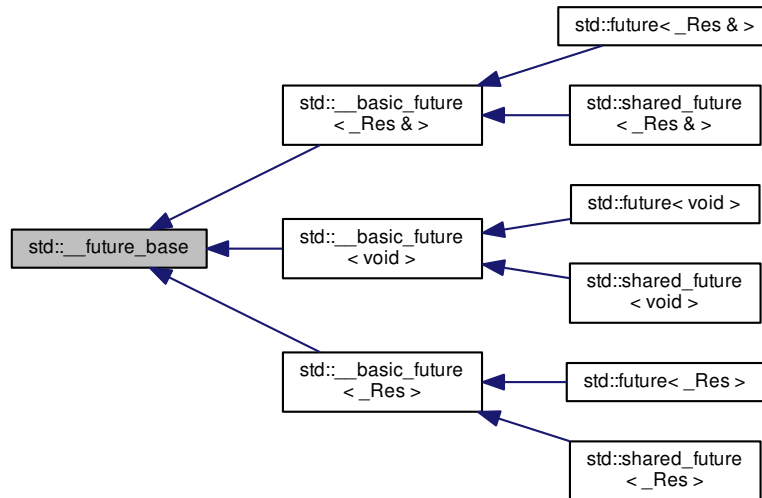
Definition at line 77 of file `exception_ptr.h`.

The documentation for this class was generated from the following file:

- [exception_ptr.h](#)

5.498 std::__future_base Struct Reference

Inheritance diagram for `std::__future_base`:



Classes

- [struct _Result](#)
- [struct _Result<_Res &>](#)
- [struct _Result<void>](#)
- [struct _Result_alloc](#)
- [struct _Result_base](#)

Public Types

- `template<typename _Res>`
`using _Ptr = unique_ptr<_Res, _Result_base::_Deleter>`
- `using _State_base = _State_baseV2`

Static Public Member Functions

- `template<typename _Res , typename _Allocator >
static _Ptr< _Result_alloc< _Res, _Allocator > > _S_allocate_result (const _Allocator &__a)`
- `template<typename _Res , typename _Tp >
static _Ptr< _Result< _Res > > _S_allocate_result (const std::allocator< _Tp > &__a)`
- `template<typename _BoundFn >
static std::shared_ptr< _State_base > _S_make_async_state (_BoundFn &&__fn)`
- `template<typename _BoundFn >
static std::shared_ptr< _State_base > _S_make_deferred_state (_BoundFn &&__fn)`
- `template<typename _Res_ptr , typename _BoundFn >
static _Task_setter< _Res_ptr, _BoundFn > _S_task_setter (_Res_ptr &__ptr, _BoundFn &__call)`

5.498.1 Detailed Description

Base class and enclosing scope.

Definition at line 189 of file future.

5.498.2 Member Typedef Documentation

5.498.2.1 `template<typename _Res > using std::__future_base::_Ptr = unique_ptr< _Res, _Result_base::_Deleter>`

A `unique_ptr` for result objects.

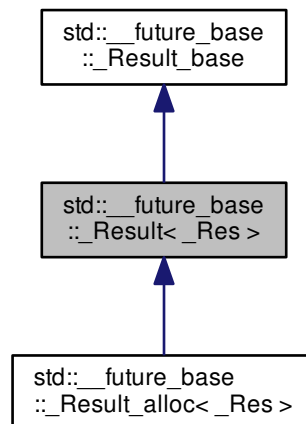
Definition at line 214 of file future.

The documentation for this struct was generated from the following file:

- [future](#)

5.499 std::__future_base::_Result< _Res > Struct Template Reference

Inheritance diagram for `std::__future_base::_Result< _Res >`:



Public Types

- typedef `_Res` **result_type**

Public Member Functions

- void **_M_set** (const `_Res` &__res)
- void **_M_set** (`_Res` &&__res)
- `_Res` & **_M_value** () noexcept

Public Attributes

- exception_ptr **_M_error**

5.499.1 Detailed Description

```
template<typename _Res>
struct std::__future_base::Result< _Res >
```

A result object that has storage for an object of type `_Res`.

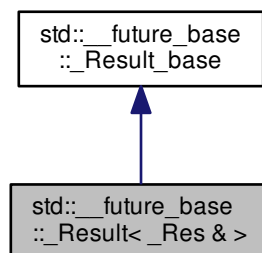
Definition at line 218 of file `future`.

The documentation for this struct was generated from the following file:

- [future](#)

5.500 `std::__future_base::Result< _Res & >` Struct Template Reference

Inheritance diagram for `std::__future_base::Result< _Res & >`:



Public Types

- typedef `_Res` & **result_type**

Public Member Functions

- `_Res` & **M_get** () noexcept
- void **M_set** (`_Res` & __res) noexcept

Public Attributes

- exception_ptr **M_error**

5.500.1 Detailed Description

```
template<typename _Res>
struct std::__future_base::_Result< _Res & >
```

Partial specialization for reference types.

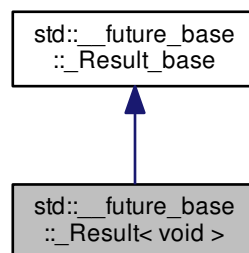
Definition at line 597 of file future.

The documentation for this struct was generated from the following file:

- [future](#)

5.501 std::__future_base::_Result< void > Struct Template Reference

Inheritance diagram for `std::__future_base::_Result< void >`:



Public Types

- typedef void **result_type**

Public Attributes

- exception_ptr **_M_error**

5.501.1 Detailed Description

```
template<>
struct std::__future_base::_Result< void >
```

Explicit specialization for void.

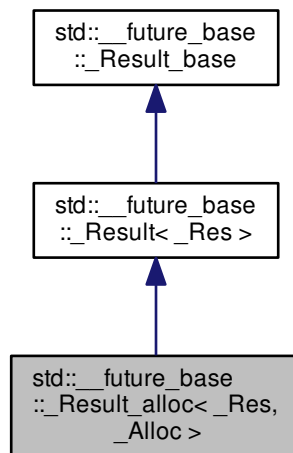
Definition at line 617 of file future.

The documentation for this struct was generated from the following file:

- [future](#)

5.502 std::__future_base::_Result_alloc< _Res, _Alloc > Struct Template Reference

Inheritance diagram for std::__future_base::_Result_alloc< _Res, _Alloc >:



Public Types

- using **__allocator_type** = __alloc_rebind< _Alloc, [_Result_alloc](#) >
- typedef _Res **result_type**

Public Member Functions

- **_Result_alloc** (const _Alloc &__a)
- void **_M_set** (const _Res &__res)
- void **_M_set** (_Res &&__res)
- _Res & **_M_value** () noexcept

Public Attributes

- exception_ptr **_M_error**

5.502.1 Detailed Description

```
template<typename _Res, typename _Alloc>
struct std::__future_base::_Result_alloc< _Res, _Alloc >
```

A result object that uses an allocator.

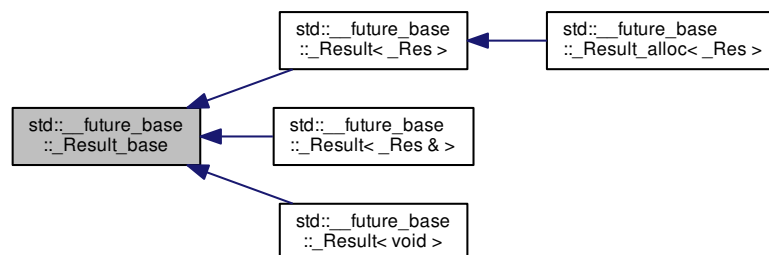
Definition at line 259 of file future.

The documentation for this struct was generated from the following file:

- [future](#)

5.503 std::__future_base::_Result_base Struct Reference

Inheritance diagram for std::__future_base::_Result_base:



Public Member Functions

- **_Result_base** (const [_Result_base](#) &)=delete
- virtual void **_M_destroy** ()=0
- [_Result_base](#) & **operator=** (const [_Result_base](#) &)=delete

Public Attributes

- exception_ptr **_M_error**

5.503.1 Detailed Description

Base class for results.

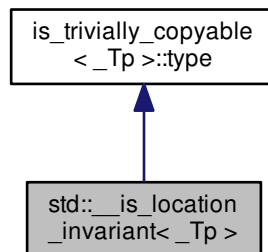
Definition at line 192 of file future.

The documentation for this struct was generated from the following file:

- [future](#)

5.504 `std::__is_location_invariant<_Tp>` Struct Template Reference

Inheritance diagram for `std::__is_location_invariant<_Tp>`:



Public Types

- typedef [integral_constant](#)<_Tp, __v> **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr _Tp **value**

5.504.1 Detailed Description

```
template<typename _Tp>  
struct std::__is_location_invariant<_Tp>
```

Trait identifying "location-invariant" types, meaning that the address of the object (or any of its members) will not escape. Trivially copyable types are location-invariant and users can specialize this trait for other types.

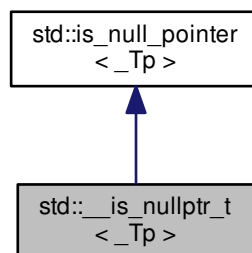
Definition at line 1442 of file functional.

The documentation for this struct was generated from the following file:

- [functional](#)

5.505 std::__is_nullptr_t<_Tp> Struct Template Reference

Inheritance diagram for std::__is_nullptr_t<_Tp>:



5.505.1 Detailed Description

```
template<typename _Tp>
struct std::__is_nullptr_t< _Tp >
```

__is_nullptr_t (extension).

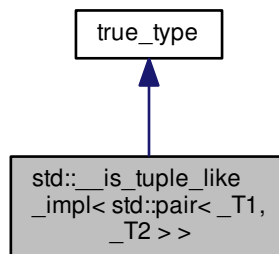
Definition at line 569 of file type_traits.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.506 std::__is_tuple_like_impl< std::pair< _T1, _T2 > > Struct Template Reference

Inheritance diagram for std::__is_tuple_like_impl< std::pair< _T1, _T2 > >:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr _Tp **value**

5.506.1 Detailed Description

```
template<typename _T1, typename _T2>
struct std::__is_tuple_like_impl< std::pair< _T1, _T2 > >
```

Partial specialization for std::pair.

Definition at line 141 of file utility.

The documentation for this struct was generated from the following file:

- [utility](#)

5.507 std::__iterator_traits<_Iterator, typename > Struct Template Reference

5.507.1 Detailed Description

```
template<typename _Iterator, typename = __void_t<>>
struct std::__iterator_traits< _Iterator, typename >
```

Traits class for iterators.

This class does nothing but define nested typedefs. The general version simply *forwards* the nested typedefs from the Iterator argument. Specialized versions for pointers and pointers-to-const provide tighter, more correct semantics.

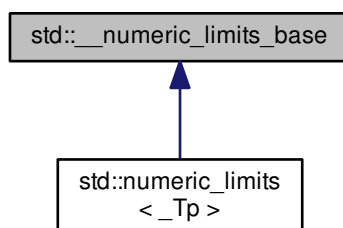
Definition at line 144 of file stl_iterator_base_types.h.

The documentation for this struct was generated from the following file:

- [stl_iterator_base_types.h](#)

5.508 std::__numeric_limits_base Struct Reference

Inheritance diagram for std::__numeric_limits_base:



Static Public Attributes

- static constexpr int [digits](#)
- static constexpr int [digits10](#)
- static constexpr [float_denorm_style](#) [has_denorm](#)
- static constexpr bool [has_denorm_loss](#)
- static constexpr bool [has_infinity](#)
- static constexpr bool [has_quiet_NaN](#)
- static constexpr bool [has_signaling_NaN](#)
- static constexpr bool [is_bounded](#)
- static constexpr bool [is_exact](#)
- static constexpr bool [is_iec559](#)
- static constexpr bool [is_integer](#)
- static constexpr bool [is_modulo](#)
- static constexpr bool [is_signed](#)
- static constexpr bool [is_specialized](#)
- static constexpr int [max_digits10](#)
- static constexpr int [max_exponent](#)
- static constexpr int [max_exponent10](#)
- static constexpr int [min_exponent](#)
- static constexpr int [min_exponent10](#)
- static constexpr int [radix](#)
- static constexpr [float_round_style](#) [round_style](#)
- static constexpr bool [tinyness_before](#)
- static constexpr bool [traps](#)

5.508.1 Detailed Description

Part of `std::numeric_limits`.

The `static const` members are usable as integral constant expressions.

Note

This is a separate class for purposes of efficiency; you should only access these members as part of an instantiation of the `std::numeric_limits` class.

Definition at line 202 of file `limits`.

5.508.2 Member Data Documentation

5.508.2.1 `constexpr int std::__numeric_limits_base::digits` `[static]`

The number of `radix` digits that be represented without change: for integer types, the number of non-sign bits in the mantissa; for floating types, the number of `radix` digits in the mantissa.

Definition at line 211 of file `limits`.

5.508.2.2 constexpr int std::__numeric_limits_base::digits10 [static]

The number of base 10 digits that can be represented without change.

Definition at line 214 of file limits.

5.508.2.3 constexpr float_denorm_style std::__numeric_limits_base::has_denorm [static]

See std::float_denorm_style for more information.

Definition at line 266 of file limits.

5.508.2.4 constexpr bool std::__numeric_limits_base::has_denorm_loss [static]

True if loss of accuracy is detected as a denormalization loss, rather than as an inexact result.

Definition at line 270 of file limits.

5.508.2.5 constexpr bool std::__numeric_limits_base::has_infinity [static]

True if the type has a representation for positive infinity.

Definition at line 255 of file limits.

5.508.2.6 constexpr bool std::__numeric_limits_base::has_quiet_NaN [static]

True if the type has a representation for a quiet (non-signaling) Not a Number.

Definition at line 259 of file limits.

5.508.2.7 constexpr bool std::__numeric_limits_base::has_signaling_NaN [static]

True if the type has a representation for a signaling Not a Number.

Definition at line 263 of file limits.

5.508.2.8 constexpr bool std::__numeric_limits_base::is_bounded [static]

True if the set of values representable by the type is finite. All built-in types are bounded, this member would be false for arbitrary precision types.

Definition at line 279 of file limits.

5.508.2.9 constexpr bool std::__numeric_limits_base::is_exact [static]

True if the type uses an exact representation. All integer types are exact, but not all exact types are integer. For example, rational and fixed-exponent representations are exact but not integer.

Definition at line 231 of file limits.

5.508.2.10 `constexpr bool std::__numeric_limits_base::is_iec559` `[static]`

True if-and-only-if the type adheres to the IEC 559 standard, also known as IEEE 754. (Only makes sense for floating point types.)

Definition at line 274 of file limits.

5.508.2.11 `constexpr bool std::__numeric_limits_base::is_integer` `[static]`

True if the type is integer.

Definition at line 226 of file limits.

5.508.2.12 `constexpr bool std::__numeric_limits_base::is_modulo` `[static]`

True if the type is *modulo*. A type is modulo if, for any operation involving `+`, `-`, or `*` on values of that type whose result would fall outside the range `[min(),max())`, the value returned differs from the true value by an integer multiple of `max() - min() + 1`. On most machines, this is false for floating types, true for unsigned integers, and true for signed integers. See PR22200 about signed integers.

Definition at line 288 of file limits.

5.508.2.13 `constexpr bool std::__numeric_limits_base::is_signed` `[static]`

True if the type is signed.

Definition at line 223 of file limits.

5.508.2.14 `constexpr bool std::__numeric_limits_base::is_specialized` `[static]`

This will be true for all fundamental types (which have specializations), and false for everything else.

Definition at line 206 of file limits.

5.508.2.15 `constexpr int std::__numeric_limits_base::max_digits10` `[static]`

The number of base 10 digits required to ensure that values which differ are always differentiated.

Definition at line 219 of file limits.

5.508.2.16 `constexpr int std::__numeric_limits_base::max_exponent` `[static]`

The maximum positive integer such that `radix` raised to the power of (one less than that integer) is a representable finite floating point number.

Definition at line 248 of file limits.

5.508.2.17 `constexpr int std::__numeric_limits_base::max_exponent10` `[static]`

The maximum positive integer such that 10 raised to that power is in the range of representable finite floating point numbers.

Definition at line 252 of file limits.

5.508.2.18 `constexpr int std::__numeric_limits_base::min_exponent` `[static]`

The minimum negative integer such that `radix` raised to the power of (one less than that integer) is a normalized floating point number.

Definition at line 239 of file limits.

5.508.2.19 `constexpr int std::__numeric_limits_base::min_exponent10` `[static]`

The minimum negative integer such that 10 raised to that power is in the range of normalized floating point numbers.

Definition at line 243 of file limits.

5.508.2.20 `constexpr int std::__numeric_limits_base::radix` `[static]`

For integer types, specifies the base of the representation. For floating types, specifies the base of the exponent representation.

Definition at line 235 of file limits.

5.508.2.21 `constexpr float_round_style std::__numeric_limits_base::round_style` `[static]`

See `std::float_round_style` for more information. This is only meaningful for floating types; integer types will all be `round_toward_zero`.

Definition at line 299 of file limits.

5.508.2.22 `constexpr bool std::__numeric_limits_base::tinyness_before` `[static]`

True if tininess is detected before rounding. (see IEC 559)

Definition at line 294 of file limits.

5.508.2.23 `constexpr bool std::__numeric_limits_base::traps` `[static]`

True if trapping is implemented for this type.

Definition at line 291 of file limits.

The documentation for this struct was generated from the following file:

- [limits](#)

5.509 `std::__parallel::_CRandNumber<_MustBeInt>` > Struct Template Reference

Public Member Functions

- `int operator() (int __limit)`

5.509.1 Detailed Description

```
template<typename _MustBeInt = int>
struct std::__parallel::_CRandNumber<_MustBeInt>
```

Functor wrapper for `std::rand()`.

Definition at line 1649 of file `algo.h`.

The documentation for this struct was generated from the following file:

- [algo.h](#)

5.510 `std::__profile::bitset<_Nb>` > Class Template Reference

Inherits `bitset<_Nb>`.

Public Member Functions

- constexpr **bitset** (unsigned long long __val) noexcept
- template<typename _CharT, typename _Traits, typename _Alloc>
 bitset (const [std::basic_string](#)<_CharT, _Traits, _Alloc> &__str, typename [std::basic_string](#)<_CharT, _Traits, _Alloc>::size_type __pos=0, typename [std::basic_string](#)<_CharT, _Traits, _Alloc>::size_type __n=([std::basic_string](#)<_CharT, _Traits, _Alloc>::npos))
- template<class _CharT, class _Traits, class _Alloc>
 bitset (const [std::basic_string](#)<_CharT, _Traits, _Alloc> &__str, typename [std::basic_string](#)<_CharT, _Traits, _Alloc>::size_type __pos, typename [std::basic_string](#)<_CharT, _Traits, _Alloc>::size_type __n, _CharT __zero, _CharT __one=_CharT('1'))
- **bitset** (const [_Base](#) &__x)
- template<typename _CharT>
 bitset (const _CharT *__str, typename [std::basic_string](#)<_CharT>::size_type __n=[std::basic_string](#)<_CharT>::npos, _CharT __zero=_CharT('0'), _CharT __one=_CharT('1'))
- [_Base](#) & [_M_base](#) () noexcept
- const [_Base](#) & [_M_base](#) () const noexcept
- [bitset](#)<_Nb> & **flip** () noexcept
- [bitset](#)<_Nb> & **flip** (size_t __pos)
- bool **operator!=** (const [bitset](#)<_Nb> &__rhs) const noexcept
- [bitset](#)<_Nb> & **operator&=** (const [bitset](#)<_Nb> &__rhs) noexcept
- [bitset](#)<_Nb> **operator<<** (size_t __pos) const noexcept
- [bitset](#)<_Nb> & **operator<<=** (size_t __pos) noexcept
- bool **operator==** (const [bitset](#)<_Nb> &__rhs) const noexcept

- [bitset](#)< _Nb > **operator**>> (size_t __pos) const noexcept
- [bitset](#)< _Nb > & **operator**>>= (size_t __pos) noexcept
- [bitset](#)< _Nb > & **operator**^= (const [bitset](#)< _Nb > &__rhs) noexcept
- [bitset](#)< _Nb > & **operator**|= (const [bitset](#)< _Nb > &__rhs) noexcept
- [bitset](#)< _Nb > **operator**~ () const noexcept
- [bitset](#)< _Nb > & **reset** () noexcept
- [bitset](#)< _Nb > & **reset** (size_t __pos)
- [bitset](#)< _Nb > & **set** () noexcept
- [bitset](#)< _Nb > & **set** (size_t __pos, bool __val=true)

5.510.1 Detailed Description

```
template<size_t _Nb>
class std::__profile::bitset< _Nb >
```

Class std::bitset wrapper with performance instrumentation, none at the moment.

Definition at line 41 of file profile/bitset.

The documentation for this class was generated from the following file:

- [profile/bitset](#)

5.511 std::__profile::deque< _Tp, _Allocator > Class Template Reference

Inherits deque< _Tp, _Allocator >.

Public Types

- typedef _Base::size_type **size_type**
- typedef _Base::value_type **value_type**

Public Member Functions

- **deque** (const [deque](#) &)=default
- **deque** ([deque](#) &&)=default
- **deque** (const [deque](#) &__d, const _Allocator &__a)
- **deque** ([deque](#) &&__d, const _Allocator &__a)
- **deque** ([initializer_list](#)< value_type > __l, const _Allocator &__a=_Allocator())
- **deque** (const _Allocator &__a)
- **deque** (size_type __n, const _Allocator &__a=_Allocator())
- **deque** (size_type __n, const _Tp &__value, const _Allocator &__a=_Allocator())
- template<typename _InputIterator, typename = std::::RequireInputIter<_InputIterator>>>
 deque (_InputIterator __first, _InputIterator __last, const _Allocator &__a=_Allocator())
- **deque** (const [_Base](#) &__x)
- [_Base](#) & **_M_base** () noexcept
- const [_Base](#) & **_M_base** () const noexcept
- [deque](#) & **operator**= (const [deque](#) &)=default
- [deque](#) & **operator**= ([deque](#) &&)=default
- [deque](#) & **operator**= ([initializer_list](#)< value_type > __l)
- void **swap** ([deque](#) &__x) noexcept(*/*conditional */*)

5.511.1 Detailed Description

```
template<typename _Tp, typename _Allocator = std::allocator<_Tp>>
class std::__profile::deque< _Tp, _Allocator >
```

Class std::deque wrapper with performance instrumentation.

Definition at line 40 of file profile/deque.

The documentation for this class was generated from the following file:

- [profile/deque](#)

5.512 std::__profile::forward_list< _Tp, _Alloc > Class Template Reference

Inherits forward_list< _Tp, _Alloc >.

Public Types

- typedef _Base::const_iterator **const_iterator**
- typedef _Base::size_type **size_type**

Public Member Functions

- **forward_list** (const _Alloc &__al) noexcept
- **forward_list** (const [forward_list](#) &__list, const _Alloc &__al)
- **forward_list** ([forward_list](#) &&__list, const _Alloc &__al)
- **forward_list** (size_type __n, const _Alloc &__al=_Alloc())
- **forward_list** (size_type __n, const _Tp &__value, const _Alloc &__al=_Alloc())
- template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>>
 forward_list (_InputIterator __first, _InputIterator __last, const _Alloc &__al=_Alloc())
- **forward_list** (const [forward_list](#) &)=default
- **forward_list** ([forward_list](#) &&)=default
- **forward_list** ([std::initializer_list](#)< _Tp > __il, const _Alloc &__al=_Alloc())
- [_Base](#) & [_M_base](#) () noexcept
- const [_Base](#) & [_M_base](#) () const noexcept
- void **merge** ([forward_list](#) &&__list)
- void **merge** ([forward_list](#) &__list)
- template<typename _Comp >
 void **merge** ([forward_list](#) &&__list, _Comp __comp)
- template<typename _Comp >
 void **merge** ([forward_list](#) &__list, _Comp __comp)
- [forward_list](#) & **operator=** (const [forward_list](#) &)=default
- [forward_list](#) & **operator=** ([forward_list](#) &&)=default
- [forward_list](#) & **operator=** ([std::initializer_list](#)< _Tp > __il)
- void **splice_after** (const_iterator __pos, [forward_list](#) &&__fl)
- void **splice_after** (const_iterator __pos, [forward_list](#) &__list)
- void **splice_after** (const_iterator __pos, [forward_list](#) &&__list, const_iterator __i)
- void **splice_after** (const_iterator __pos, [forward_list](#) &__list, const_iterator __i)
- void **splice_after** (const_iterator __pos, [forward_list](#) &&__list, const_iterator __before, const_iterator __last)
- void **splice_after** (const_iterator __pos, [forward_list](#) &__list, const_iterator __before, const_iterator __last)
- void **swap** ([forward_list](#) &__fl) noexcept(noexcept(declval< [_Base](#) & >().swap(__fl)))

5.512.1 Detailed Description

```
template<typename _Tp, typename _Alloc = std::allocator<_Tp>>
class std::__profile::forward_list< _Tp, _Alloc >
```

Class std::forward_list wrapper with performance instrumentation.

Definition at line 44 of file profile/forward_list.

The documentation for this class was generated from the following file:

- [profile/forward_list](#)

5.513 std::__profile::list< _Tp, _Allocator > Class Template Reference

Inherits list< _Tp, _Allocator >, and std::__profile::_List_profile< _List >.

Public Types

- typedef _Allocator **allocator_type**
- typedef __iterator_tracker< typename _Base::const_iterator, [list](#) > **const_iterator**
- typedef _Base::const_pointer **const_pointer**
- typedef _Base::const_reference **const_reference**
- typedef [std::reverse_iterator](#)< const_iterator > **const_reverse_iterator**
- typedef _Base::difference_type **difference_type**
- typedef __iterator_tracker< typename _Base::iterator, [list](#) > **iterator**
- typedef _Base::pointer **pointer**
- typedef _Base::reference **reference**
- typedef [std::reverse_iterator](#)< iterator > **reverse_iterator**
- typedef _Base::size_type **size_type**
- typedef _Tp **value_type**

Public Member Functions

- **list** (const [list](#) &)=default
- **list** ([list](#) &&)=default
- **list** ([initializer_list](#)< value_type > __l, const allocator_type &__a=allocator_type())
- **list** (const [list](#) &__x, const allocator_type &__a)
- **list** ([list](#) &&__x, const allocator_type &__a)
- **list** (const _Allocator &__a) noexcept
- **list** (size_type __n, const allocator_type &__a=allocator_type())
- **list** (size_type __n, const _Tp &__value, const _Allocator &__a=_Allocator())
- template<typename _InputIterator , typename = std::_RequireInputIter<_InputIterator>>>
 list (_InputIterator __first, _InputIterator __last, const _Allocator &__a=_Allocator())
- **list** (const [_Base](#) &__x)
- [_Base](#) & **_M_base** () noexcept
- const [_Base](#) & **_M_base** () const noexcept

- void **_M_profile_construct** () noexcept
- void **_M_profile_destruct** () noexcept
- void **_M_profile_iterate** (int __rewind=0) const
- void **_M_swap** (_List_profile &__other)
- reference **back** () noexcept
- const_reference **back** () const noexcept
- iterator **begin** () noexcept
- const_iterator **begin** () const noexcept
- const_iterator **cbegin** () const noexcept
- const_iterator **cend** () const noexcept
- void **clear** () noexcept
- [const_reverse_iterator](#) **crbegin** () const noexcept
- [const_reverse_iterator](#) **crend** () const noexcept
- template<typename... _Args>
iterator **emplace** (const_iterator __position, _Args &&... __args)
- iterator **end** () noexcept
- const_iterator **end** () const noexcept
- iterator **erase** (const_iterator __pos) noexcept
- iterator **erase** (const_iterator __pos, const_iterator __last) noexcept
- iterator **insert** (const_iterator __pos, const_Tp &__x)
- iterator **insert** (const_iterator __pos, _Tp && __x)
- iterator **insert** (const_iterator __pos, [initializer_list](#)< value_type > __l)
- iterator **insert** (const_iterator __pos, size_type __n, const_Tp &__x)
- template<typename _InputIterator, typename = std::::RequireInputIter<_InputIterator>>>
iterator **insert** (const_iterator __pos, _InputIterator __first, _InputIterator __last)
- void **merge** ([list](#) && __x)
- void **merge** ([list](#) & __x)
- template<class _Compare >
void **merge** ([list](#) && __x, _Compare __comp)
- template<typename _Compare >
void **merge** ([list](#) & __x, _Compare __comp)
- [list](#) & **operator=** (const [list](#) &)=default
- [list](#) & **operator=** ([list](#) &&)=default
- [list](#) & **operator=** ([initializer_list](#)< value_type > __l)
- void **pop_back** () noexcept
- void **pop_front** () noexcept
- void **push_front** (const value_type &__x)
- [reverse_iterator](#) **rbegin** () noexcept
- [const_reverse_iterator](#) **rbegin** () const noexcept
- void **remove** (const_Tp &__value)
- template<class _Predicate >
void **remove_if** (_Predicate __pred)
- [reverse_iterator](#) **rend** () noexcept
- [const_reverse_iterator](#) **rend** () const noexcept
- void **splice** (const_iterator __pos, [list](#) && __x) noexcept
- void **splice** (const_iterator __pos, [list](#) & __x) noexcept
- void **splice** (const_iterator __pos, [list](#) & __x, const_iterator __i)
- void **splice** (const_iterator __pos, [list](#) && __x, const_iterator __i) noexcept
- void **splice** (const_iterator __pos, [list](#) && __x, const_iterator __first, const_iterator __last) noexcept
- void **splice** (const_iterator __pos, [list](#) & __x, const_iterator __first, const_iterator __last) noexcept
- void **swap** ([list](#) & __x) noexcept(/*conditional */)
- void **unique** ()
- template<class _BinaryPredicate >
void **unique** (_BinaryPredicate __binary_pred)

Public Attributes

- `__gnu_profile::__list2slist_info * _M_list2slist_info`
- `__gnu_profile::__list2vector_info * _M_list2vector_info`

5.513.1 Detailed Description

```
template<typename _Tp, typename _Allocator = std::allocator<_Tp>>
class std::__profile::list<_Tp, _Allocator>
```

List wrapper with performance instrumentation.

Definition at line 106 of file `profile/list`.

The documentation for this class was generated from the following file:

- [profile/list](#)

5.514 `std::__profile::map<_Key, _Tp, _Compare, _Allocator>` Class Template Reference

Inherits `map<_Key, _Tp, _Compare, _Allocator>`, and `std::__profile::__Ordered_profile<_Cont>`.

Public Types

- `typedef __iterator_tracker<_Base_const_iterator, map> const_iterator`
- `typedef _Base::const_reference const_reference`
- `typedef std::reverse_iterator<const_iterator> const_reverse_iterator`
- `typedef _Base::difference_type difference_type`
- `typedef __iterator_tracker<_Base_iterator, map> iterator`
- `typedef _Compare key_compare`
- `typedef _Key key_type`
- `typedef _Tp mapped_type`
- `typedef _Base::reference reference`
- `typedef std::reverse_iterator<iterator> reverse_iterator`
- `typedef _Base::size_type size_type`
- `typedef _Base::value_type value_type`

Public Member Functions

- **map** (const **map** &)=default
- **map** (**map** &&)=default
- **map** (const **_Compare** &__comp, const **_Allocator** &__a= **_Allocator**())
- template<typename **_InputIterator** , typename = std::_RequireInputIter<_InputIterator>>>
map (**_InputIterator** __first, **_InputIterator** __last, const **_Compare** &__comp= **_Compare**(), const **_Allocator** &__a= **_Allocator**())
- **map** (const **_Base** &__x)
- **map** (**initializer_list**< value_type > __l, const **_Compare** &__c= **_Compare**(), const **_Allocator** &__a= **_Allocator**())
- **map** (const **_Allocator** &__a)
- **map** (const **map** &__x, const **_Allocator** &__a)
- **map** (**map** &&__x, const **_Allocator** &__a) noexcept(noexcept(**_Base**(std::move(__x), __a)))
- **map** (**initializer_list**< value_type > __l, const **_Allocator** &__a)
- template<typename **_InputIterator** >
map (**_InputIterator** __first, **_InputIterator** __last, const **_Allocator** &__a)
- **_Base** & **_M_base** () noexcept
- const **_Base** & **_M_base** () const noexcept
- void **_M_profile_iterate** (int __rewind=0) const
- mapped_type & **at** (const key_type &__k)
- const mapped_type & **at** (const key_type &__k) const
- iterator **begin** () noexcept
- const_iterator **begin** () const noexcept
- const_iterator **cbegin** () const noexcept
- const_iterator **cend** () const noexcept
- void **clear** () noexcept
- size_type **count** (const key_type &__x) const
- template<typename **_Kt** , typename **_Req** = typename __has_is_transparent<_Compare, **_Kt**::type>
size_type **count** (const **_Kt** &__x) const
- **const_reverse_iterator** **crbegin** () const noexcept
- **const_reverse_iterator** **crend** () const noexcept
- template<typename... **_Args**>
std::pair< iterator, bool > **emplace** (**_Args** &&... __args)
- template<typename... **_Args**>
iterator **emplace_hint** (const_iterator __pos, **_Args** &&... __args)
- iterator **end** () noexcept
- const_iterator **end** () const noexcept
- **std::pair**< iterator, iterator > **equal_range** (const key_type &__x)
- template<typename **_Kt** , typename **_Req** = typename __has_is_transparent<_Compare, **_Kt**::type>
std::pair< iterator, iterator > **equal_range** (const **_Kt** &__x)
- **std::pair**< const_iterator, const_iterator > **equal_range** (const key_type &__x) const
- template<typename **_Kt** , typename **_Req** = typename __has_is_transparent<_Compare, **_Kt**::type>
std::pair< const_iterator, const_iterator > **equal_range** (const **_Kt** &__x) const
- iterator **erase** (const_iterator __pos)
- iterator **erase** (iterator __pos)
- size_type **erase** (const key_type &__x)
- iterator **erase** (const_iterator __first, const_iterator __last)
- iterator **find** (const key_type &__x)
- template<typename **_Kt** , typename **_Req** = typename __has_is_transparent<_Compare, **_Kt**::type>
iterator **find** (const **_Kt** &__x)
- const_iterator **find** (const key_type &__x) const

- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
const_iterator **find** (const _Kt &__x) const
- [std::pair](#)< iterator, bool > **insert** (const value_type &__x)
- template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>
[std::pair](#)< iterator, bool > **insert** (_Pair &&__x)
- void **insert** ([std::initializer_list](#)< value_type > __list)
- iterator **insert** (const_iterator __pos, const value_type &__x)
- template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>
iterator **insert** (const_iterator __pos, _Pair &&__x)
- template<typename _InputIterator >
void **insert** (_InputIterator __first, _InputIterator __last)
- iterator **lower_bound** (const key_type &__x)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
iterator **lower_bound** (const _Kt &__x)
- const_iterator **lower_bound** (const key_type &__x) const
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
const_iterator **lower_bound** (const _Kt &__x) const
- [map](#) & **operator=** (const [map](#) &)=default
- [map](#) & **operator=** ([map](#) &&)=default
- [map](#) & **operator=** ([initializer_list](#)< value_type > __l)
- mapped_type & **operator[]** (const key_type &__k)
- mapped_type & **operator[]** (key_type &&__k)
- [reverse_iterator](#) **rbegin** () noexcept
- [const_reverse_iterator](#) **rbegin** () const noexcept
- [reverse_iterator](#) **rend** () noexcept
- [const_reverse_iterator](#) **rend** () const noexcept
- void **swap** ([map](#) &__x) noexcept(*/*conditional */*)
- iterator **upper_bound** (const key_type &__x)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
iterator **upper_bound** (const _Kt &__x)
- const_iterator **upper_bound** (const key_type &__x) const
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
const_iterator **upper_bound** (const _Kt &__x) const

Protected Member Functions

- void **_M_profile_construct** () noexcept
- void **_M_profile_destruct** () noexcept
- void **_M_swap** (_Ordered_profile &__other)

Protected Attributes

- [__gnu_profile::__map2umap_info](#) * **_M_map2umap_info**

Friends

- template<typename _K1, typename _T1, typename _C1, typename _A1 >
bool **operator<** (const [map](#)< _K1, _T1, _C1, _A1 > &, const [map](#)< _K1, _T1, _C1, _A1 > &)
- template<typename _K1, typename _T1, typename _C1, typename _A1 >
bool **operator==** (const [map](#)< _K1, _T1, _C1, _A1 > &, const [map](#)< _K1, _T1, _C1, _A1 > &)

5.514.1 Detailed Description

```
template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Allocator = std::allocator<std::pair<const _Key, _Tp>>>
class std::__profile::map<_Key, _Tp, _Compare, _Allocator>
```

Class std::map wrapper with performance instrumentation.

Definition at line 41 of file profile/map.h.

The documentation for this class was generated from the following file:

- [profile/map.h](#)

5.515 std::__profile::multimap< _Key, _Tp, _Compare, _Allocator > Class Template Reference

Inherits multimap< _Key, _Tp, _Compare, _Allocator >, and std::__profile::_Ordered_profile< _Cont >.

Public Types

- typedef __iterator_tracker< _Base_const_iterator, [multimap](#) > **const_iterator**
- typedef _Base::const_reference **const_reference**
- typedef [std::reverse_iterator](#)< const_iterator > **const_reverse_iterator**
- typedef _Base::difference_type **difference_type**
- typedef __iterator_tracker< _Base_iterator, [multimap](#) > **iterator**
- typedef _Compare **key_compare**
- typedef _Key **key_type**
- typedef _Tp **mapped_type**
- typedef _Base::reference **reference**
- typedef [std::reverse_iterator](#)< iterator > **reverse_iterator**
- typedef _Base::size_type **size_type**
- typedef [std::pair](#)< const _Key, _Tp > **value_type**

Public Member Functions

- **multimap** (const [multimap](#) &)=default
- **multimap** ([multimap](#) &&)=default
- **multimap** (const _Compare &__comp, const _Allocator &__a=_Allocator())
- template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>>
 multimap (_InputIterator __first, _InputIterator __last, const _Compare &__comp=_Compare(), const _Allocator &__a=_Allocator())
- **multimap** ([initializer_list](#)< [value_type](#) > __l, const _Compare &__c=_Compare(), const _Allocator &__a=_Allocator())
- **multimap** (const _Allocator &__a)
- **multimap** (const [multimap](#) &__x, const _Allocator &__a)
- **multimap** ([multimap](#) &&__x, const _Allocator &__a) noexcept(noexcept([_Base](#)(std::move(__x), __a)))
- **multimap** ([initializer_list](#)< [value_type](#) > __l, const _Allocator &__a)

- template<typename _InputIterator >
 multimap (_InputIterator __first, _InputIterator __last, const _Allocator &__a)
- **multimap** (const _Base &__x)
- _Base & _M_base () noexcept
- const _Base & _M_base () const noexcept
- void _M_profile_iterate (int __rewind=0) const
- iterator **begin** () noexcept
- const_iterator **begin** () const noexcept
- const_iterator **cbegin** () const noexcept
- const_iterator **cend** () const noexcept
- void **clear** () noexcept
- size_type **count** (const key_type &__x) const
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
 size_type **count** (const _Kt &__x) const
- const_reverse_iterator **crbegin** () const noexcept
- const_reverse_iterator **crend** () const noexcept
- template<typename... _Args>
 iterator **emplace** (_Args &&... __args)
- template<typename... _Args>
 iterator **emplace_hint** (const_iterator __pos, _Args &&... __args)
- iterator **end** () noexcept
- const_iterator **end** () const noexcept
- std::pair< iterator, iterator > **equal_range** (const key_type &__x)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
 std::pair< iterator, iterator > **equal_range** (const _Kt &__x)
- std::pair< const_iterator, const_iterator > **equal_range** (const key_type &__x) const
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
 std::pair< const_iterator, const_iterator > **equal_range** (const _Kt &__x) const
- iterator **erase** (const_iterator __pos)
- iterator **erase** (iterator __pos)
- size_type **erase** (const key_type &__x)
- iterator **erase** (const_iterator __first, const_iterator __last)
- iterator **find** (const key_type &__x)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
 iterator **find** (const _Kt &__x)
- const_iterator **find** (const key_type &__x) const
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
 const_iterator **find** (const _Kt &__x) const
- iterator **insert** (const value_type &__x)
- template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>
 iterator **insert** (_Pair && __x)
- void **insert** (std::initializer_list< value_type > __list)
- iterator **insert** (const_iterator __pos, const value_type &__x)
- template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>
 iterator **insert** (const_iterator __pos, _Pair && __x)
- template<typename _InputIterator >
 void **insert** (_InputIterator __first, _InputIterator __last)
- iterator **lower_bound** (const key_type &__x)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
 iterator **lower_bound** (const _Kt &__x)
- const_iterator **lower_bound** (const key_type &__x) const

- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`const_iterator lower_bound (const _Kt &__x) const`
- `multimap & operator= (const multimap &)=default`
- `multimap & operator= (multimap &&)=default`
- `multimap & operator= (initializer_list< value_type > __l)`
- `reverse_iterator rbegin () noexcept`
- `const_reverse_iterator rbegin () const noexcept`
- `reverse_iterator rend () noexcept`
- `const_reverse_iterator rend () const noexcept`
- `void swap (multimap &__x) noexcept(/*conditional */)`
- `iterator upper_bound (const key_type &__x)`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`iterator upper_bound (const _Kt &__x)`
- `const_iterator upper_bound (const key_type &__x) const`
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
`const_iterator upper_bound (const _Kt &__x) const`

Protected Member Functions

- `void _M_profile_construct () noexcept`
- `void _M_profile_destruct () noexcept`
- `void _M_swap (_Ordered_profile &__other)`

Protected Attributes

- `__gnu_profile::__map2umap_info * _M_map2umap_info`

Friends

- `template<typename _K1, typename _T1, typename _C1, typename _A1 >`
`bool operator< (const multimap< _K1, _T1, _C1, _A1 > &, const multimap< _K1, _T1, _C1, _A1 > &)`
- `template<typename _K1, typename _T1, typename _C1, typename _A1 >`
`bool operator== (const multimap< _K1, _T1, _C1, _A1 > &, const multimap< _K1, _T1, _C1, _A1 > &)`

5.515.1 Detailed Description

```
template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Allocator = std::allocator<std::pair<const _Key, _Tp>>>
class std::__profile::multimap< _Key, _Tp, _Compare, _Allocator >
```

Class `std::multimap` wrapper with performance instrumentation.

Definition at line 42 of file `profile/multimap.h`.

The documentation for this class was generated from the following file:

- [profile/multimap.h](#)

5.516 std::__profile::multiset< _Key, _Compare, _Allocator > Class Template Reference

Inherits multiset< _Key, _Compare, _Allocator >, and std::__profile::_Ordered_profile< _Cont >.

Public Types

- typedef _Allocator **allocator_type**
- typedef __iterator_tracker< _Base_const_iterator, [multiset](#) > **const_iterator**
- typedef _Base::const_reference **const_reference**
- typedef [std::reverse_iterator](#)< const_iterator > **const_reverse_iterator**
- typedef _Base::difference_type **difference_type**
- typedef __iterator_tracker< _Base_iterator, [multiset](#) > **iterator**
- typedef _Compare **key_compare**
- typedef _Key **key_type**
- typedef _Base::reference **reference**
- typedef [std::reverse_iterator](#)< iterator > **reverse_iterator**
- typedef _Base::size_type **size_type**
- typedef _Compare **value_compare**
- typedef _Key **value_type**

Public Member Functions

- **multiset** (const [multiset](#) &)=default
- **multiset** ([multiset](#) &&)=default
- **multiset** (const _Compare &__comp, const _Allocator &__a=_Allocator())
- template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>>
multiset (_InputIterator __first, _InputIterator __last, const _Compare &__comp=_Compare(), const _Allocator &__a=_Allocator())
- **multiset** ([initializer_list](#)< value_type > __l, const _Compare &__comp=_Compare(), const allocator_type &__a=allocator_type())
- **multiset** (const allocator_type &__a)
- **multiset** (const [multiset](#) &__x, const allocator_type &__a)
- **multiset** ([multiset](#) &&__x, const allocator_type &__a) noexcept(noexcept([_Base](#)(std::move(__x), __a)))
- **multiset** ([initializer_list](#)< value_type > __l, const allocator_type &__a)
- template<typename _InputIterator >
multiset (_InputIterator __first, _InputIterator __last, const allocator_type &__a)
- **multiset** (const [_Base](#) &__x)
- [_Base](#) & **_M_base** () noexcept
- const [_Base](#) & **_M_base** () const noexcept
- void **_M_profile_iterate** (int __rewind=0) const
- iterator **begin** () noexcept
- const_iterator **begin** () const noexcept
- const_iterator **cbegin** () const noexcept
- const_iterator **cend** () const noexcept
- void **clear** () noexcept
- size_type **count** (const key_type &__x) const
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
size_type **count** (const _Kt &__x) const
- [const_reverse_iterator](#) **crbegin** () const noexcept

- [const_reverse_iterator](#) **crend** () const noexcept
- `template<typename... _Args>`
iterator **emplace** (_Args &&... __args)
- `template<typename... _Args>`
iterator **emplace_hint** (const_iterator __pos, _Args &&... __args)
- iterator **end** () noexcept
- const_iterator **end** () const noexcept
- [std::pair](#)< iterator, iterator > **equal_range** (const key_type &__x)
- [std::pair](#)< const_iterator, const_iterator > **equal_range** (const key_type &__x) const
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
[std::pair](#)< iterator, iterator > **equal_range** (const _Kt &__x)
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
[std::pair](#)< const_iterator, const_iterator > **equal_range** (const _Kt &__x) const
- iterator **erase** (const_iterator __pos)
- size_type **erase** (const key_type &__x)
- iterator **erase** (const_iterator __first, const_iterator __last)
- iterator **find** (const key_type &__x)
- const_iterator **find** (const key_type &__x) const
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
iterator **find** (const _Kt &__x)
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
const_iterator **find** (const _Kt &__x) const
- iterator **insert** (const value_type &__x)
- iterator **insert** (value_type &&__x)
- iterator **insert** (const_iterator __pos, const value_type &__x)
- iterator **insert** (const_iterator __pos, value_type &&__x)
- `template<typename _InputIterator, typename = std::::RequireInputIter<_InputIterator>>>`
void **insert** (_InputIterator __first, _InputIterator __last)
- void **insert** ([initializer_list](#)< value_type > __l)
- iterator **lower_bound** (const key_type &__x)
- const_iterator **lower_bound** (const key_type &__x) const
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
iterator **lower_bound** (const _Kt &__x)
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
const_iterator **lower_bound** (const _Kt &__x) const
- [multiset](#) & **operator=** (const [multiset](#) &)=default
- [multiset](#) & **operator=** ([multiset](#) &&)=default
- [multiset](#) & **operator=** ([initializer_list](#)< value_type > __l)
- [reverse_iterator](#) **rbegin** () noexcept
- [const_reverse_iterator](#) **rbegin** () const noexcept
- [reverse_iterator](#) **rend** () noexcept
- [const_reverse_iterator](#) **rend** () const noexcept
- void **swap** ([multiset](#) &__x) noexcept(*/*conditional */*)
- iterator **upper_bound** (const key_type &__x)
- const_iterator **upper_bound** (const key_type &__x) const
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
iterator **upper_bound** (const _Kt &__x)
- `template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>`
const_iterator **upper_bound** (const _Kt &__x) const

Protected Member Functions

- void **_M_profile_construct** () noexcept
- void **_M_profile_destruct** () noexcept
- void **_M_swap** (_Ordered_profile &__other)

Protected Attributes

- [__gnu_profile::__map2umap_info](#) * **_M_map2umap_info**

Friends

- template<typename _K1, typename _C1, typename _A1 >
bool **operator**< (const [multiset](#)< _K1, _C1, _A1 > &, const [multiset](#)< _K1, _C1, _A1 > &)
- template<typename _K1, typename _C1, typename _A1 >
bool **operator==** (const [multiset](#)< _K1, _C1, _A1 > &, const [multiset](#)< _K1, _C1, _A1 > &)

5.516.1 Detailed Description

```
template<typename _Key, typename _Compare = std::less<_Key>, typename _Allocator = std::allocator<_Key>>
class std::__profile::multiset< _Key, _Compare, _Allocator >
```

Class std::multiset wrapper with performance instrumentation.

Definition at line 42 of file profile/multiset.h.

The documentation for this class was generated from the following file:

- [profile/multiset.h](#)

5.517 std::__profile::set< _Key, _Compare, _Allocator > Class Template Reference

Inherits set< _Key, _Compare, _Allocator >, and std::__profile::__Ordered_profile< _Cont >.

Public Types

- typedef __iterator_tracker< _Base_const_iterator, [set](#) > **const_iterator**
- typedef _Base::const_reference **const_reference**
- typedef [std::reverse_iterator](#)< const_iterator > **const_reverse_iterator**
- typedef _Base::difference_type **difference_type**
- typedef __iterator_tracker< _Base_iterator, [set](#) > **iterator**
- typedef _Compare **key_compare**
- typedef _Key **key_type**
- typedef _Base::reference **reference**
- typedef [std::reverse_iterator](#)< iterator > **reverse_iterator**
- typedef _Base::size_type **size_type**
- typedef _Compare **value_compare**
- typedef _Key **value_type**

Public Member Functions

- **set** (const [set](#) &)=default
- **set** ([set](#) &&)=default
- **set** (const [_Compare](#) &__comp, const [_Allocator](#) &__a=[_Allocator](#)())
- template<typename [_InputIterator](#) , typename = std::RequireInputIter<[_InputIterator](#)>>
set ([_InputIterator](#) __first, [_InputIterator](#) __last, const [_Compare](#) &__comp=[_Compare](#)(), const [_Allocator](#) &__a=[_Allocator](#)())
- **set** ([initializer_list](#)< [value_type](#) > __l, const [_Compare](#) &__comp=[_Compare](#)(), const [_Allocator](#) &__a=[_Allocator](#)())
- **set** (const [_Allocator](#) &__a)
- **set** (const [set](#) &__x, const [_Allocator](#) &__a)
- **set** ([set](#) &&__x, const [_Allocator](#) &__a) noexcept(noexcept([_Base](#)(std::move(__x), __a)))
- **set** ([initializer_list](#)< [value_type](#) > __l, const [_Allocator](#) &__a)
- template<typename [_InputIterator](#) >
set ([_InputIterator](#) __first, [_InputIterator](#) __last, const [_Allocator](#) &__a)
- **set** (const [_Base](#) &__x)
- [_Base](#) & [_M_base](#) () noexcept
- const [_Base](#) & [_M_base](#) () const noexcept
- void [_M_profile_iterate](#) (int __rewind=0) const
- iterator **begin** () noexcept
- const_iterator **begin** () const noexcept
- const_iterator **cbegin** () const noexcept
- const_iterator **cend** () const noexcept
- void **clear** () noexcept
- size_type **count** (const key_type &__x) const
- template<typename [_Kt](#) , typename [_Req](#) = typename [__has_is_transparent](#)<[_Compare](#), [_Kt](#)>::type>
size_type **count** (const [_Kt](#) &__x) const
- [const_reverse_iterator](#) **crbegin** () const noexcept
- [const_reverse_iterator](#) **crend** () const noexcept
- template<typename... [_Args](#)>
[std::pair](#)< iterator, bool > **emplace** ([_Args](#) &&...__args)
- template<typename... [_Args](#)>
iterator **emplace_hint** (const_iterator __pos, [_Args](#) &&...__args)
- iterator **end** () noexcept
- const_iterator **end** () const noexcept
- [std::pair](#)< iterator, iterator > **equal_range** (const key_type &__x)
- [std::pair](#)< const_iterator, const_iterator > **equal_range** (const key_type &__x) const
- template<typename [_Kt](#) , typename [_Req](#) = typename [__has_is_transparent](#)<[_Compare](#), [_Kt](#)>::type>
[std::pair](#)< iterator, iterator > **equal_range** (const [_Kt](#) &__x)
- template<typename [_Kt](#) , typename [_Req](#) = typename [__has_is_transparent](#)<[_Compare](#), [_Kt](#)>::type>
[std::pair](#)< const_iterator, const_iterator > **equal_range** (const [_Kt](#) &__x) const
- iterator **erase** (const_iterator __pos)
- size_type **erase** (const key_type &__x)
- iterator **erase** (const_iterator __first, const_iterator __last)
- iterator **find** (const key_type &__x)
- const_iterator **find** (const key_type &__x) const
- template<typename [_Kt](#) , typename [_Req](#) = typename [__has_is_transparent](#)<[_Compare](#), [_Kt](#)>::type>
iterator **find** (const [_Kt](#) &__x)
- template<typename [_Kt](#) , typename [_Req](#) = typename [__has_is_transparent](#)<[_Compare](#), [_Kt](#)>::type>
const_iterator **find** (const [_Kt](#) &__x) const

- `std::pair< iterator, bool > insert` (const value_type &__x)
- `std::pair< iterator, bool > insert` (value_type &&__x)
- iterator `insert` (const_iterator __pos, const value_type &__x)
- iterator `insert` (const_iterator __pos, value_type &&__x)
- template<typename _InputIterator >
void `insert` (_InputIterator __first, _InputIterator __last)
- void `insert` (initializer_list< value_type > __l)
- iterator `lower_bound` (const key_type &__x)
- const_iterator `lower_bound` (const key_type &__x) const
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
iterator `lower_bound` (const _Kt &__x)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
const_iterator `lower_bound` (const _Kt &__x) const
- `set` & `operator=` (const `set` &)=default
- `set` & `operator=` (`set` &&)=default
- `set` & `operator=` (initializer_list< value_type > __l)
- `reverse_iterator` `rbegin` () noexcept
- `const_reverse_iterator` `rbegin` () const noexcept
- `reverse_iterator` `rend` () noexcept
- `const_reverse_iterator` `rend` () const noexcept
- void `swap` (`set` &__x) noexcept(*conditional *)
- iterator `upper_bound` (const key_type &__x)
- const_iterator `upper_bound` (const key_type &__x) const
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
iterator `upper_bound` (const _Kt &__x)
- template<typename _Kt, typename _Req = typename __has_is_transparent<_Compare, _Kt>::type>
const_iterator `upper_bound` (const _Kt &__x) const

Protected Member Functions

- void `_M_profile_construct` () noexcept
- void `_M_profile_destruct` () noexcept
- void `_M_swap` (_Ordered_profile &__other)

Protected Attributes

- `__gnu_profile::__map2umap_info` * `_M_map2umap_info`

Friends

- template<typename _K1, typename _C1, typename _A1 >
bool `operator<` (const `set`< _K1, _C1, _A1 > &, const `set`< _K1, _C1, _A1 > &)
- template<typename _K1, typename _C1, typename _A1 >
bool `operator==` (const `set`< _K1, _C1, _A1 > &, const `set`< _K1, _C1, _A1 > &)

5.517.1 Detailed Description

```
template<typename _Key, typename _Compare = std::less<_Key>, typename _Allocator = std::allocator<_Key>>
class std::__profile::set< _Key, _Compare, _Allocator >
```

Class std::set wrapper with performance instrumentation.

Definition at line 42 of file profile/set.h.

The documentation for this class was generated from the following file:

- [profile/set.h](#)

5.518 std::__profile::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > Class Template Reference

Inherits unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, and std::__profile::_Unordered_profile< _Unordered< Cont, _Unique_keys >.

Public Types

- typedef _Base::allocator_type **allocator_type**
- typedef _Base::const_iterator **const_iterator**
- typedef _Base::const_reference **const_reference**
- typedef _Base::difference_type **difference_type**
- typedef _Base::hasher **hasher**
- typedef _Base::iterator **iterator**
- typedef _Base::key_equal **key_equal**
- typedef _Base::key_type **key_type**
- typedef _Base::mapped_type **mapped_type**
- typedef _Base::reference **reference**
- typedef _Base::size_type **size_type**
- typedef _Base::value_type **value_type**

Public Member Functions

- **unordered_map** (size_type __n, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- template<typename _InputIterator >
unordered_map (_InputIterator __f, _InputIterator __l, size_type __n=0, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- **unordered_map** (const [unordered_map](#) &)=default
- **unordered_map** (const [_Base](#) &__x)
- **unordered_map** ([unordered_map](#) &&)=default
- **unordered_map** (const allocator_type &__a)
- **unordered_map** (const [unordered_map](#) &__umap, const allocator_type &__a)
- **unordered_map** ([unordered_map](#) &&__umap, const allocator_type &__a)

- **unordered_map** ([initializer_list](#)< value_type > __l, size_type __n=0, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- **unordered_map** (size_type __n, const allocator_type &__a)
- **unordered_map** (size_type __n, const hasher &__hf, const allocator_type &__a)
- template<typename _InputIterator >
unordered_map (_InputIterator __first, _InputIterator __last, size_type __n, const allocator_type &__a)
- template<typename _InputIterator >
unordered_map (_InputIterator __first, _InputIterator __last, size_type __n, const hasher &__hf, const allocator_type &__a)
- **unordered_map** ([initializer_list](#)< value_type > __l, size_type __n, const allocator_type &__a)
- **unordered_map** ([initializer_list](#)< value_type > __l, size_type __n, const hasher &__hf, const allocator_type &__a)
- void **clear** () noexcept
- template<typename... _Args>
[std::pair](#)< iterator, bool > **emplace** (_Args &&...__args)
- template<typename... _Args>
iterator **emplace_hint** (const_iterator __it, _Args &&...__args)
- void **insert** ([std::initializer_list](#)< value_type > __l)
- [std::pair](#)< iterator, bool > **insert** (const value_type &__obj)
- iterator **insert** (const_iterator __iter, const value_type &__v)
- template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>
[std::pair](#)< iterator, bool > **insert** (_Pair &&__obj)
- template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>
iterator **insert** (const_iterator __iter, _Pair &&__v)
- template<typename _InputIter >
void **insert** (_InputIter __first, _InputIter __last)
- **unordered_map** & **operator=** (const **unordered_map** &)=default
- **unordered_map** & **operator=** (**unordered_map** &&)=default
- **unordered_map** & **operator=** ([initializer_list](#)< value_type > __l)
- mapped_type & **operator[]** (const _Key &__k)
- mapped_type & **operator[]** (_Key &&__k)
- void **rehash** (size_type __n)
- void **swap** (**unordered_map** &__x) noexcept(noexcept(__x._M_base().swap(__x)))

Protected Member Functions

- void **_M_profile_construct** () noexcept
- void **_M_profile_destruct** () noexcept
- void **_M_profile_resize** (std::size_t __old_size)
- void **_M_swap** (_Unordered_profile &__other) noexcept

Protected Attributes

- [__gnu_profile::__hashfunc_info](#) * **_M_hashfunc_info**
- [__gnu_profile::__container_size_info](#) * **_M_size_info**

5.518.1 Detailed Description

```
template<typename _Key, typename _Tp, typename _Hash = std::hash<_Key>, typename _Pred = std::equal_to<_Key>, typename
_Alloc = std::allocator<std::pair<const _Key, _Tp> >>
class std::__profile::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >
```

Class std::unordered_map wrapper with performance instrumentation.

Definition at line 51 of file profile/unordered_map.

The documentation for this class was generated from the following file:

- [profile/unordered_map](#)

5.519 std::__profile::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > Class Template Reference

Inherits unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, and std::__profile::Unordered_profile< _Key, _Tp, _Hash, _Pred, _Alloc >, and std::__profile::Unordered_profile< _Key, _Tp, _Hash, _Pred, _Alloc >, and std::__profile::Unordered_profile< _Key, _Tp, _Hash, _Pred, _Alloc >, and std::__profile::Unordered_profile< _Key, _Tp, _Hash, _Pred, _Alloc >.

Public Types

- typedef _Base::allocator_type **allocator_type**
- typedef _Base::const_iterator **const_iterator**
- typedef _Base::const_reference **const_reference**
- typedef _Base::difference_type **difference_type**
- typedef _Base::hasher **hasher**
- typedef _Base::iterator **iterator**
- typedef _Base::key_equal **key_equal**
- typedef _Base::key_type **key_type**
- typedef _Base::reference **reference**
- typedef _Base::size_type **size_type**
- typedef _Base::value_type **value_type**

Public Member Functions

- **unordered_multimap** (size_type __n, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- template<typename _InputIterator >
unordered_multimap (_InputIterator __f, _InputIterator __l, size_type __n=0, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- **unordered_multimap** (const [unordered_multimap](#) &)=default
- **unordered_multimap** (const [_Base](#) &__x)
- **unordered_multimap** ([unordered_multimap](#) &&)=default
- **unordered_multimap** (const allocator_type &__a)
- **unordered_multimap** (const [unordered_multimap](#) &__ummap, const allocator_type &__a)
- **unordered_multimap** ([unordered_multimap](#) &&__ummap, const allocator_type &__a)

- **unordered_multimap** ([initializer_list](#)< value_type > __l, size_type __n=0, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- **unordered_multimap** (size_type __n, const allocator_type &__a)
- **unordered_multimap** (size_type __n, const hasher &__hf, const allocator_type &__a)
- template<typename _InputIterator >
unordered_multimap (_InputIterator __first, _InputIterator __last, size_type __n, const allocator_type &__a)
- template<typename _InputIterator >
unordered_multimap (_InputIterator __first, _InputIterator __last, size_type __n, const hasher &__hf, const allocator_type &__a)
- **unordered_multimap** ([initializer_list](#)< value_type > __l, size_type __n, const allocator_type &__a)
- **unordered_multimap** ([initializer_list](#)< value_type > __l, size_type __n, const hasher &__hf, const allocator_type &__a)
- void **clear** () noexcept
- template<typename... _Args>
iterator **emplace** (_Args &&... __args)
- template<typename... _Args>
iterator **emplace_hint** (const_iterator __it, _Args &&... __args)
- void **insert** ([std::initializer_list](#)< value_type > __l)
- iterator **insert** (const value_type &__obj)
- iterator **insert** (const_iterator __iter, const value_type &__v)
- template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>
iterator **insert** (_Pair &&__obj)
- template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>
iterator **insert** (const_iterator __iter, _Pair &&__v)
- template<typename _InputIter >
void **insert** (_InputIter __first, _InputIter __last)
- **unordered_multimap** & **operator=** (const **unordered_multimap** &)=default
- **unordered_multimap** & **operator=** (**unordered_multimap** &&)=default
- **unordered_multimap** & **operator=** ([initializer_list](#)< value_type > __l)
- void **rehash** (size_type __n)
- void **swap** (**unordered_multimap** &__x) noexcept(noexcept(__x._M_base().swap(__x)))

Protected Member Functions

- void **_M_profile_construct** () noexcept
- void **_M_profile_destruct** () noexcept
- void **_M_profile_resize** (std::size_t __old_size)
- void **_M_swap** (_Unordered_profile &__other) noexcept

Protected Attributes

- [__gnu_profile::__hashfunc_info](#) * **_M_hashfunc_info**
- [__gnu_profile::__container_size_info](#) * **_M_size_info**

5.519.1 Detailed Description

```
template<typename _Key, typename _Tp, typename _Hash = std::hash<_Key>, typename _Pred = std::equal_to<_Key>, typename
_Alloc = std::allocator<std::pair<const _Key, _Tp>>>
class std::__profile::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >
```

Class std::unordered_multimap wrapper with performance instrumentation.

Definition at line 329 of file profile/unordered_map.

The documentation for this class was generated from the following file:

- [profile/unordered_map](#)

5.520 std::__profile::unordered_multiset< _Value, _Hash, _Pred, _Alloc > Class Template Reference

Inherits unordered_multiset< _Value, _Hash, _Pred, _Alloc >, and std::__profile::_Unordered_profile< _Unordered< Cont, _Unique_keys >.

Public Types

- typedef _Base::allocator_type **allocator_type**
- typedef _Base::const_iterator **const_iterator**
- typedef _Base::const_reference **const_reference**
- typedef _Base::difference_type **difference_type**
- typedef _Base::hasher **hasher**
- typedef _Base::iterator **iterator**
- typedef _Base::key_equal **key_equal**
- typedef _Base::key_type **key_type**
- typedef _Base::reference **reference**
- typedef _Base::size_type **size_type**
- typedef _Base::value_type **value_type**

Public Member Functions

- **unordered_multiset** (size_type __n, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- template<typename _InputIterator >
unordered_multiset (_InputIterator __f, _InputIterator __l, size_type __n=0, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- **unordered_multiset** (const [unordered_multiset](#) &)=default
- **unordered_multiset** (const [_Base](#) &__x)
- **unordered_multiset** ([unordered_multiset](#) &&)=default
- **unordered_multiset** (const allocator_type &__a)
- **unordered_multiset** (const [unordered_multiset](#) &__umset, const allocator_type &__a)
- **unordered_multiset** ([unordered_multiset](#) &&__umset, const allocator_type &__a)

- **unordered_multiset** ([initializer_list](#)< value_type > __l, size_type __n=0, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- **unordered_multiset** (size_type __n, const allocator_type &__a)
- **unordered_multiset** (size_type __n, const hasher &__hf, const allocator_type &__a)
- `template<typename _InputIterator >`
unordered_multiset (_InputIterator __first, _InputIterator __last, size_type __n, const allocator_type &__a)
- `template<typename _InputIterator >`
unordered_multiset (_InputIterator __first, _InputIterator __last, size_type __n, const hasher &__hf, const allocator_type &__a)
- **unordered_multiset** ([initializer_list](#)< value_type > __l, size_type __n, const allocator_type &__a)
- **unordered_multiset** ([initializer_list](#)< value_type > __l, size_type __n, const hasher &__hf, const allocator_type &__a)
- `void clear ()` noexcept
- `template<typename... _Args>`
iterator **emplace** (_Args &&... __args)
- `template<typename... _Args>`
iterator **emplace_hint** (const_iterator __it, _Args &&... __args)
- `void insert (std::initializer_list< value_type > __l)`
- `iterator insert` (const value_type &__obj)
- `iterator insert` (const_iterator __iter, const value_type &__v)
- `iterator insert` (value_type && __obj)
- `iterator insert` (const_iterator __iter, value_type && __v)
- `template<typename _InputIter >`
`void insert` (_InputIter __first, _InputIter __last)
- **unordered_multiset** & **operator=** (const **unordered_multiset** &)=default
- **unordered_multiset** & **operator=** (**unordered_multiset** &&)=default
- **unordered_multiset** & **operator=** ([initializer_list](#)< value_type > __l)
- `void rehash` (size_type __n)
- `void swap` (**unordered_multiset** &__x) noexcept(noexcept(__x._M_base().swap(__x)))

Protected Member Functions

- `void _M_profile_construct ()` noexcept
- `void _M_profile_destruct ()` noexcept
- `void _M_profile_resize` (std::size_t __old_size)
- `void _M_swap` (_Unordered_profile &__other) noexcept

Protected Attributes

- [__gnu_profile::__hashfunc_info](#) * **_M_hashfunc_info**
- [__gnu_profile::__container_size_info](#) * **_M_size_info**

5.520.1 Detailed Description

`template<typename _Value, typename _Hash = std::hash<_Value>, typename _Pred = std::equal_to<_Value>, typename _Alloc = std::allocator<_Value>>`

`class std::__profile::unordered_multiset< _Value, _Hash, _Pred, _Alloc >`

Unordered_multiset wrapper with performance instrumentation.

Definition at line 307 of file `profile/unordered_set`.

The documentation for this class was generated from the following file:

- [profile/unordered_set](#)

5.521 `std::__profile::unordered_set<_Key, _Hash, _Pred, _Alloc>` Class Template Reference

Inherits `unordered_set<_Key, _Hash, _Pred, _Alloc>`, and `std::__profile::_Unordered_profile<_UnorderedCont, _Key, _Hash, _Pred, _Alloc, Unique_keys>`.

Public Types

- `typedef _Base::allocator_type allocator_type`
- `typedef _Base::const_iterator const_iterator`
- `typedef _Base::const_reference const_reference`
- `typedef _Base::difference_type difference_type`
- `typedef _Base::hasher hasher`
- `typedef _Base::iterator iterator`
- `typedef _Base::key_equal key_equal`
- `typedef _Base::key_type key_type`
- `typedef _Base::reference reference`
- `typedef _Base::size_type size_type`
- `typedef _Base::value_type value_type`

Public Member Functions

- **`unordered_set`** (`size_type __n`, `const hasher &__hf=hasher()`, `const key_equal &__eq=key_equal()`, `const allocator_type &__a=allocator_type()`)
- `template<typename _InputIterator>`
`unordered_set` (`_InputIterator __f`, `_InputIterator __l`, `size_type __n=0`, `const hasher &__hf=hasher()`, `const key_equal &__eq=key_equal()`, `const allocator_type &__a=allocator_type()`)
- **`unordered_set`** (`const unordered_set &`)=default
- **`unordered_set`** (`const _Base &__x`)
- **`unordered_set`** (`unordered_set &&`)=default
- **`unordered_set`** (`const allocator_type &__a`)
- **`unordered_set`** (`const unordered_set &__uset`, `const allocator_type &__a`)
- **`unordered_set`** (`unordered_set &&__uset`, `const allocator_type &__a`)
- **`unordered_set`** (`initializer_list<value_type> __l`, `size_type __n=0`, `const hasher &__hf=hasher()`, `const key_equal &__eq=key_equal()`, `const allocator_type &__a=allocator_type()`)
- **`unordered_set`** (`size_type __n`, `const allocator_type &__a`)
- **`unordered_set`** (`size_type __n`, `const hasher &__hf`, `const allocator_type &__a`)
- `template<typename _InputIterator>`
`unordered_set` (`_InputIterator __first`, `_InputIterator __last`, `size_type __n`, `const allocator_type &__a`)
- `template<typename _InputIterator>`
`unordered_set` (`_InputIterator __first`, `_InputIterator __last`, `size_type __n`, `const hasher &__hf`, `const allocator_type &__a`)
- **`unordered_set`** (`initializer_list<value_type> __l`, `size_type __n`, `const allocator_type &__a`)
- **`unordered_set`** (`initializer_list<value_type> __l`, `size_type __n`, `const hasher &__hf`, `const allocator_type &__a`)
- `void clear ()` noexcept
- `template<typename... _Args>`
`std::pair<iterator, bool> emplace (_Args &&... __args)`
- `template<typename... _Args>`
`iterator emplace_hint (const_iterator __it, _Args &&... __args)`
- `void insert (std::initializer_list<value_type> __l)`

- `std::pair`< iterator, bool > **insert** (const value_type &__obj)
- iterator **insert** (const_iterator __iter, const value_type &__v)
- `std::pair`< iterator, bool > **insert** (value_type &&__obj)
- iterator **insert** (const_iterator __iter, value_type &&__v)
- template<typename _InputIter >
void **insert** (_InputIter __first, _InputIter __last)
- `unordered_set` & **operator=** (const `unordered_set` &)=default
- `unordered_set` & **operator=** (`unordered_set` &&)=default
- `unordered_set` & **operator=** (initializer_list< value_type > __l)
- void **rehash** (size_type __n)
- void **swap** (`unordered_set` &__x) noexcept(noexcept(__x._M_base().swap(__x)))

Protected Member Functions

- void **_M_profile_construct** () noexcept
- void **_M_profile_destruct** () noexcept
- void **_M_profile_resize** (std::size_t __old_size)
- void **_M_swap** (_Unordered_profile &__other) noexcept

Protected Attributes

- `__gnu_profile::__hashfunc_info` * **_M_hashfunc_info**
- `__gnu_profile::__container_size_info` * **_M_size_info**

5.521.1 Detailed Description

```
template<typename _Key, typename _Hash = std::hash<_Key>, typename _Pred = std::equal_to<_Key>, typename _Alloc = std::allocator<_Key>>
class std::__profile::unordered_set<_Key,_Hash,_Pred,_Alloc>
```

`Unordered_set` wrapper with performance instrumentation.

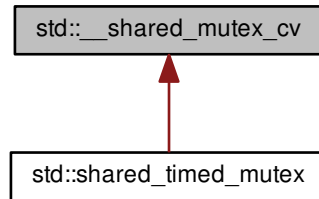
Definition at line 51 of file `profile/unordered_set`.

The documentation for this class was generated from the following file:

- [profile/unordered_set](#)

5.522 `std::__shared_mutex_cv` Class Reference

Inheritance diagram for `std::__shared_mutex_cv`:



Public Member Functions

- `__shared_mutex_cv` (const `__shared_mutex_cv` &)=delete
- void **lock** ()
- void **lock_shared** ()
- `__shared_mutex_cv` & **operator=** (const `__shared_mutex_cv` &)=delete
- bool **try_lock** ()
- bool **try_lock_shared** ()
- void **unlock** ()
- void **unlock_shared** ()

Friends

- class **shared_timed_mutex**

5.522.1 Detailed Description

A shared mutex type implemented using `std::condition_variable`.

Definition at line 174 of file `shared_mutex`.

The documentation for this class was generated from the following file:

- [shared_mutex](#)

5.523 `std::_Base_bitset<_Nw>` Struct Template Reference

Public Types

- typedef unsigned long **_WordT**

Public Member Functions

- constexpr **_Base_bitset** (unsigned long long __val) noexcept
- template<size_t _Nb>
bool **_M_are_all** () const noexcept
- void **_M_do_and** (const [_Base_bitset](#)<_Nw> &__x) noexcept
- size_t **_M_do_count** () const noexcept
- size_t **_M_do_find_first** (size_t) const noexcept
- size_t **_M_do_find_next** (size_t, size_t) const noexcept
- void **_M_do_flip** () noexcept
- void **_M_do_left_shift** (size_t __shift) noexcept
- void **_M_do_or** (const [_Base_bitset](#)<_Nw> &__x) noexcept
- void **_M_do_reset** () noexcept
- void **_M_do_right_shift** (size_t __shift) noexcept
- void **_M_do_set** () noexcept
- unsigned long long **_M_do_to_ullong** () const
- unsigned long **_M_do_to_ulong** () const
- void **_M_do_xor** (const [_Base_bitset](#)<_Nw> &__x) noexcept
- const _WordT * **_M_getdata** () const noexcept
- _WordT & **_M_getword** (size_t __pos) noexcept
- constexpr _WordT **_M_getword** (size_t __pos) const noexcept
- _WordT & **_M_hiword** () noexcept
- constexpr _WordT **_M_hiword** () const noexcept
- bool **_M_is_any** () const noexcept
- bool **_M_is_equal** (const [_Base_bitset](#)<_Nw> &__x) const noexcept

Static Public Member Functions

- static constexpr _WordT **_S_maskbit** (size_t __pos) noexcept
- static constexpr size_t **_S_whichbit** (size_t __pos) noexcept
- static constexpr size_t **_S_whichbyte** (size_t __pos) noexcept
- static constexpr size_t **_S_whichword** (size_t __pos) noexcept

Public Attributes

- _WordT **_M_w** [_Nw]

5.523.1 Detailed Description

```
template<size_t _Nw>
struct std::_Base_bitset<_Nw>
```

Base class, general case. It is a class invariant that _Nw will be nonnegative.

See documentation for bitset.

Definition at line 71 of file bitset.

5.523.2 Member Data Documentation

5.523.2.1 `template<size_t _Nw> _WordT std::_Base_bitset<_Nw>::_M_w[_Nw]`

0 is the least significant word.

Definition at line 76 of file `bitset`.

The documentation for this struct was generated from the following file:

- [bitset](#)

5.524 `std::_Base_bitset<0>` Struct Template Reference

Public Types

- typedef unsigned long `_WordT`

Public Member Functions

- constexpr `_Base_bitset` (unsigned long long) noexcept
- template<size_t _Nb>
 bool `_M_are_all` () const noexcept
- void `_M_do_and` (const [_Base_bitset](#)<0> &) noexcept
- size_t `_M_do_count` () const noexcept
- size_t `_M_do_find_first` (size_t) const noexcept
- size_t `_M_do_find_next` (size_t, size_t) const noexcept
- void `_M_do_flip` () noexcept
- void `_M_do_left_shift` (size_t) noexcept
- void `_M_do_or` (const [_Base_bitset](#)<0> &) noexcept
- void `_M_do_reset` () noexcept
- void `_M_do_right_shift` (size_t) noexcept
- void `_M_do_set` () noexcept
- unsigned long long `_M_do_to_ullong` () const noexcept
- unsigned long `_M_do_to_ulong` () const noexcept
- void `_M_do_xor` (const [_Base_bitset](#)<0> &) noexcept
- `_WordT` & `_M_getword` (size_t) noexcept
- constexpr `_WordT` `_M_getword` (size_t __pos) const noexcept
- constexpr `_WordT` `_M_hiword` () const noexcept
- bool `_M_is_any` () const noexcept
- bool `_M_is_equal` (const [_Base_bitset](#)<0> &) const noexcept

Static Public Member Functions

- static constexpr `_WordT` `_S_maskbit` (size_t __pos) noexcept
- static constexpr size_t `_S_whichbit` (size_t __pos) noexcept
- static constexpr size_t `_S_whichbyte` (size_t __pos) noexcept
- static constexpr size_t `_S_whichword` (size_t __pos) noexcept

5.524.1 Detailed Description

```
template<>
struct std::_Base_bitset< 0 >
```

Base class, specialization for no storage (zero-length bitset).

See documentation for bitset.

Definition at line 519 of file bitset.

The documentation for this struct was generated from the following file:

- [bitset](#)

5.525 std::_Base_bitset< 1 > Struct Template Reference

Public Types

- typedef unsigned long **_WordT**

Public Member Functions

- constexpr **_Base_bitset** (unsigned long long __val) noexcept
- template<size_t _Nb>
bool **_M_are_all** () const noexcept
- void **_M_do_and** (const [_Base_bitset](#)< 1 > &__x) noexcept
- size_t **_M_do_count** () const noexcept
- size_t **_M_do_find_first** (size_t __not_found) const noexcept
- size_t **_M_do_find_next** (size_t __prev, size_t __not_found) const noexcept
- void **_M_do_flip** () noexcept
- void **_M_do_left_shift** (size_t __shift) noexcept
- void **_M_do_or** (const [_Base_bitset](#)< 1 > &__x) noexcept
- void **_M_do_reset** () noexcept
- void **_M_do_right_shift** (size_t __shift) noexcept
- void **_M_do_set** () noexcept
- unsigned long long **_M_do_to_ullong** () const noexcept
- unsigned long **_M_do_to_ulong** () const noexcept
- void **_M_do_xor** (const [_Base_bitset](#)< 1 > &__x) noexcept
- const _WordT * **_M_getdata** () const noexcept
- _WordT & **_M_getword** (size_t) noexcept
- constexpr _WordT **_M_getword** (size_t) const noexcept
- _WordT & **_M_hiword** () noexcept
- constexpr _WordT **_M_hiword** () const noexcept
- bool **_M_is_any** () const noexcept
- bool **_M_is_equal** (const [_Base_bitset](#)< 1 > &__x) const noexcept

Static Public Member Functions

- static constexpr `_WordT _S_maskbit` (`size_t __pos`) noexcept
- static constexpr `size_t _S_whichbit` (`size_t __pos`) noexcept
- static constexpr `size_t _S_whichbyte` (`size_t __pos`) noexcept
- static constexpr `size_t _S_whichword` (`size_t __pos`) noexcept

Public Attributes

- `_WordT _M_w`

5.525.1 Detailed Description

```
template<>
struct std::_Base_bitset< 1 >
```

Base class, specialization for a single word.

See documentation for `bitset`.

Definition at line 372 of file `bitset`.

The documentation for this struct was generated from the following file:

- [bitset](#)

5.526 `std::_Bind< _Signature >` Struct Template Reference

5.526.1 Detailed Description

```
template<typename _Signature>
struct std::_Bind< _Signature >
```

Type of the function object returned from `bind()`.

Definition at line 915 of file `functional`.

The documentation for this struct was generated from the following file:

- [functional](#)

5.527 std::_Bind_result< _Result, _Signature > Struct Template Reference

5.527.1 Detailed Description

```
template<typename _Result, typename _Signature>
struct std::_Bind_result< _Result, _Signature >
```

Type of the function object returned from bind<R>().

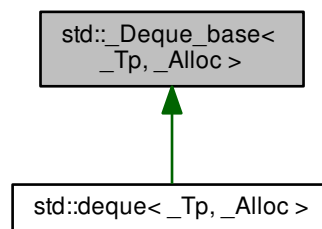
Definition at line 1041 of file functional.

The documentation for this struct was generated from the following file:

- [functional](#)

5.528 std::_Deque_base< _Tp, _Alloc > Class Template Reference

Inheritance diagram for std::_Deque_base< _Tp, _Alloc >:



Public Types

- typedef _Alloc **allocator_type**
- typedef [_Deque_iterator](#)< _Tp, const _Tp &, _Ptr_const > **const_iterator**
- typedef [_Deque_iterator](#)< _Tp, _Tp &, _Ptr > **iterator**
- typedef _Alloc_traits::size_type **size_type**

Public Member Functions

- **_Deque_base** (size_t __num_elements)
- **_Deque_base** (const allocator_type &__a, size_t __num_elements)
- **_Deque_base** (const allocator_type &__a)
- **_Deque_base** ([_Deque_base](#) &&__x, [false_type](#))
- **_Deque_base** ([_Deque_base](#) &&__x, [true_type](#))
- **_Deque_base** ([_Deque_base](#) &&__x)
- **_Deque_base** ([_Deque_base](#) &&__x, const allocator_type &__a, size_type __n)
- allocator_type **get_allocator** () const noexcept

Protected Types

- enum { **_S_initial_map_size** }
- typedef `__gnu_cxx::__alloc_traits<_Tp_alloc_type>` **_Alloc_traits**
- typedef `__gnu_cxx::__alloc_traits<_Map_alloc_type>` **_Map_alloc_traits**
- typedef `_Alloc_traits::template rebind<_Ptr>::other` **_Map_alloc_type**
- typedef `iterator::_Map_pointer` **_Map_pointer**
- typedef `_Alloc_traits::pointer` **_Ptr**
- typedef `_Alloc_traits::const_pointer` **_Ptr_const**
- typedef `__gnu_cxx::__alloc_traits<_Alloc>::template rebind<_Tp>::other` **_Tp_alloc_type**

Protected Member Functions

- `_Map_pointer` **_M_allocate_map** (size_t __n)
- `_Ptr` **_M_allocate_node** ()
- void **_M_create_nodes** (_Map_pointer __nstart, _Map_pointer __nfinish)
- void **_M_deallocate_map** (_Map_pointer __p, size_t __n) noexcept
- void **_M_deallocate_node** (_Ptr __p) noexcept
- void **_M_destroy_nodes** (_Map_pointer __nstart, _Map_pointer __nfinish) noexcept
- `_Map_alloc_type` **_M_get_map_allocator** () const noexcept
- `_Tp_alloc_type &` **_M_get_Tp_allocator** () noexcept
- const `_Tp_alloc_type &` **_M_get_Tp_allocator** () const noexcept
- void **_M_initialize_map** (size_t)

Protected Attributes

- `_Deque_impl` **_M_impl**

5.528.1 Detailed Description

```
template<typename _Tp, typename _Alloc>
class std::_Deque_base<_Tp, _Alloc>
```

Deque base class. This class provides the unified face for deque's allocation. This class's constructor and destructor allocate and deallocate (but do not initialize) storage. This makes exception safety easier.

Nothing in this class ever constructs or destroys an actual `Tp` element. (Deque handles that itself.) Only/All memory management is performed here.

Definition at line 458 of file `stl_deque.h`.

5.528.2 Member Function Documentation

5.528.2.1 `template<typename _Tp, typename _Alloc> void std::_Deque_base<_Tp, _Alloc>::_M_initialize_map (size_t __num_elements)` [protected]

Layout storage.

Parameters

<code>__num_elements</code>	The count of T's for which to allocate space at first.
-----------------------------	--

Returns

Nothing.

The initial underlying memory layout is a bit complicated...

Definition at line 680 of file `stl_deque.h`.

References `std::max()`.

The documentation for this class was generated from the following file:

- [stl_deque.h](#)

5.529 `std::_Deque_iterator< _Tp, _Ref, _Ptr >` Struct Template Reference

Public Types

- `typedef __ptr_to< _Tp > Elt_pointer`
- `typedef __ptr_to< _Elt_pointer > Map_pointer`
- `typedef _Deque_iterator Self`
- `typedef __iter< const _Tp > const_iterator`
- `typedef ptrdiff_t difference_type`
- `typedef __iter< _Tp > iterator`
- `typedef std::random_access_iterator_tag iterator_category`
- `typedef _Ptr pointer`
- `typedef _Ref reference`
- `typedef size_t size_type`
- `typedef _Tp value_type`

Public Member Functions

- `_Deque_iterator (_Elt_pointer __x, _Map_pointer __y) noexcept`
- `_Deque_iterator (const iterator &__x) noexcept`
- `iterator _M_const_cast () const noexcept`
- `void _M_set_node (_Map_pointer __new_node) noexcept`
- `reference operator* () const noexcept`
- `_Self operator+ (difference_type __n) const noexcept`
- `_Self & operator++ () noexcept`
- `_Self operator++ (int) noexcept`
- `_Self & operator+= (difference_type __n) noexcept`
- `_Self operator- (difference_type __n) const noexcept`
- `_Self & operator-- () noexcept`
- `_Self operator-- (int) noexcept`
- `_Self & operator-= (difference_type __n) noexcept`
- `pointer operator-> () const noexcept`
- `reference operator[] (difference_type __n) const noexcept`

Static Public Member Functions

- static `size_t _S_buffer_size ()` noexcept

Public Attributes

- `_Elt_pointer _M_cur`
- `_Elt_pointer _M_first`
- `_Elt_pointer _M_last`
- `_Map_pointer _M_node`

5.529.1 Detailed Description

```
template<typename _Tp, typename _Ref, typename _Ptr>
struct std::_Deque_iterator< _Tp, _Ref, _Ptr >
```

A deque::iterator.

Quite a bit of intelligence here. Much of the functionality of deque is actually passed off to this class. A deque holds two of these internally, marking its valid range. Access to elements is done as offsets of either of those two, relying on operator overloading in this class.

All the functions are op overloads except for `_M_set_node`.

Definition at line 106 of file `stl_deque.h`.

5.529.2 Member Function Documentation

5.529.2.1 `template<typename _Tp, typename _Ref, typename _Ptr> void std::_Deque_iterator< _Tp, _Ref, _Ptr >::_M_set_node (_Map_pointer __new_node)` [inline], [noexcept]

Prepares to traverse `new_node`. Sets everything except `_M_cur`, which should therefore be set by the caller immediately afterwards, based on `_M_first` and `_M_last`.

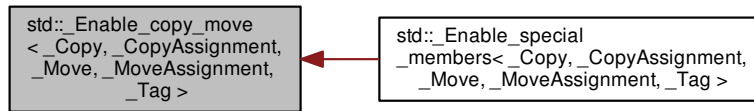
Definition at line 252 of file `stl_deque.h`.

The documentation for this struct was generated from the following file:

- [stl_deque.h](#)

5.530 `std::_Enable_copy_move< _Copy, _CopyAssignment, _Move, _MoveAssignment, _Tag >` Struct Template Reference

Inheritance diagram for `std::_Enable_copy_move< _Copy, _CopyAssignment, _Move, _MoveAssignment, _Tag >`:



5.530.1 Detailed Description

```
template<bool _Copy, bool _CopyAssignment, bool _Move, bool _MoveAssignment, typename _Tag = void>
struct std::_Enable_copy_move< _Copy, _CopyAssignment, _Move, _MoveAssignment, _Tag >
```

A mixin helper to conditionally enable or disable the copy/move special members.

See also

`_Enable_special_members`

Definition at line 64 of file `enable_special_members.h`.

The documentation for this struct was generated from the following file:

- [enable_special_members.h](#)

5.531 `std::_Enable_default_constructor< _Switch, _Tag >` Struct Template Reference

5.531.1 Detailed Description

```
template<bool _Switch, typename _Tag = void>
struct std::_Enable_default_constructor< _Switch, _Tag >
```

A mixin helper to conditionally enable or disable the default constructor.

See also

`_Enable_special_members`

Definition at line 45 of file `enable_special_members.h`.

The documentation for this struct was generated from the following file:

- [enable_special_members.h](#)

5.532 `std::_Enable_destructor<_Switch,_Tag>` Struct Template Reference

5.532.1 Detailed Description

```
template<bool _Switch, typename _Tag = void>
struct std::_Enable_destructor<_Switch,_Tag>
```

A mixin helper to conditionally enable or disable the default destructor.

See also

`_Enable_special_members`

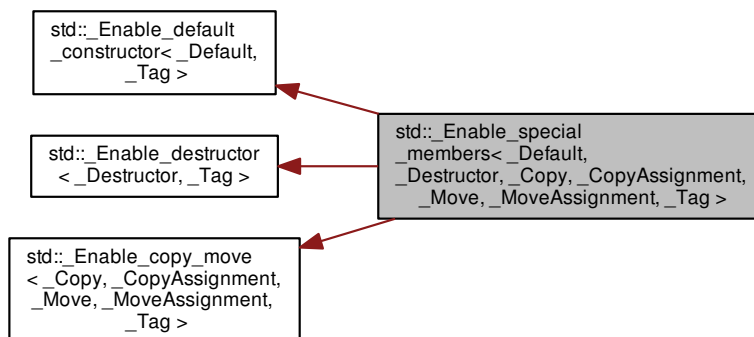
Definition at line 54 of file `enable_special_members.h`.

The documentation for this struct was generated from the following file:

- [enable_special_members.h](#)

5.533 `std::_Enable_special_members<_Default,_Destructor,_Copy,_CopyAssignment,_Move,_MoveAssignment,_Tag>` Struct Template Reference

Inheritance diagram for `std::_Enable_special_members<_Default,_Destructor,_Copy,_CopyAssignment,_Move,_MoveAssignment,_Tag>`:



5.533.1 Detailed Description

```
template<bool _Default, bool _Destructor, bool _Copy, bool _CopyAssignment, bool _Move, bool _MoveAssignment, typename _Tag =
void>
struct std::_Enable_special_members< _Default, _Destructor, _Copy, _CopyAssignment, _Move, _MoveAssignment, _Tag >
```

A mixin helper to conditionally enable or disable the special members.

The `_Tag` type parameter is to make mixin bases unique and thus avoid ambiguities.

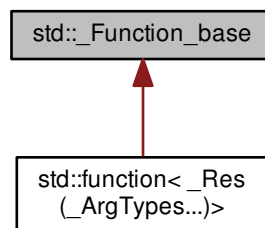
Definition at line 77 of file `enable_special_members.h`.

The documentation for this struct was generated from the following file:

- [enable_special_members.h](#)

5.534 std::_Function_base Class Reference

Inheritance diagram for `std::_Function_base`:



Public Types

- `typedef bool(* _Manager_type) (_Any_data &, const _Any_data &, _Manager_operation)`

Public Member Functions

- `bool _M_empty () const`

Public Attributes

- `_Any_data _M_functor`
- `_Manager_type _M_manager`

Static Public Attributes

- static const std::size_t **_M_max_align**
- static const std::size_t **_M_max_size**

5.534.1 Detailed Description

Base class of all polymorphic function object wrappers.

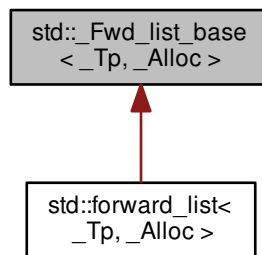
Definition at line 1529 of file functional.

The documentation for this class was generated from the following file:

- [functional](#)

5.535 std::_Fwd_list_base<_Tp, _Alloc> Struct Template Reference

Inheritance diagram for std::_Fwd_list_base<_Tp, _Alloc>:



Public Types

- typedef [_Fwd_list_node](#)<_Tp> **_Node**
- typedef [_Fwd_list_const_iterator](#)<_Tp> **const_iterator**
- typedef [_Fwd_list_iterator](#)<_Tp> **iterator**

Public Member Functions

- **_Fwd_list_base** ([_Node_alloc_type](#) &&__a)
- **_Fwd_list_base** ([_Fwd_list_base](#) &&__lst, [_Node_alloc_type](#) &&__a)
- **_Fwd_list_base** ([_Fwd_list_base](#) &&__lst)
- [_Node_alloc_type](#) & **_M_get_Node_allocator** () noexcept
- const [_Node_alloc_type](#) & **_M_get_Node_allocator** () const noexcept

Protected Types

- typedef [__gnu_cxx::__alloc_traits](#)< _Node_alloc_type > **_Node_alloc_traits**
- typedef [__alloc_rebind](#)< _Alloc, [_Fwd_list_node](#)< _Tp > > **_Node_alloc_type**
- typedef [__alloc_rebind](#)< _Alloc, _Tp > **_Tp_alloc_type**

Protected Member Functions

- template<typename... _Args>
[_Node](#) * **_M_create_node** (_Args &&... __args)
- [_Fwd_list_node_base](#) * **_M_erase_after** ([_Fwd_list_node_base](#) * __pos)
- [_Fwd_list_node_base](#) * **_M_erase_after** ([_Fwd_list_node_base](#) * __pos, [_Fwd_list_node_base](#) * __last)
- [_Node](#) * **_M_get_node** ()
- template<typename... _Args>
[_Fwd_list_node_base](#) * **_M_insert_after** (const_iterator __pos, _Args &&... __args)
- void **_M_put_node** ([_Node](#) * __p)

Protected Attributes

- [_Fwd_list_impl](#) **_M_impl**

5.535.1 Detailed Description

```
template<typename _Tp, typename _Alloc>
struct std::_Fwd_list_base< _Tp, _Alloc >
```

Base class for forward_list.

Definition at line 274 of file forward_list.h.

The documentation for this struct was generated from the following files:

- [forward_list.h](#)
- [forward_list.tcc](#)

5.536 std::_Fwd_list_const_iterator< _Tp > Struct Template Reference

Public Types

- typedef const [_Fwd_list_node](#)< _Tp > **_Node**
- typedef [_Fwd_list_const_iterator](#)< _Tp > **_Self**
- typedef ptrdiff_t **difference_type**
- typedef [_Fwd_list_iterator](#)< _Tp > **iterator**
- typedef [std::forward_iterator_tag](#) **iterator_category**
- typedef const _Tp * **pointer**
- typedef const _Tp & **reference**
- typedef _Tp **value_type**

Public Member Functions

- [_Fwd_list_const_iterator](#) (const [_Fwd_list_node_base](#) * __n) noexcept
- [_Fwd_list_const_iterator](#) (const [iterator](#) & __iter) noexcept
- [_Self](#) [_M_next](#) () const noexcept
- bool **operator!=** (const [_Self](#) & __x) const noexcept
- reference **operator*** () const noexcept
- [_Self](#) & **operator++** () noexcept
- [_Self](#) **operator++** (int) noexcept
- pointer **operator->** () const noexcept
- bool **operator==** (const [_Self](#) & __x) const noexcept

Public Attributes

- const [_Fwd_list_node_base](#) * [_M_node](#)

5.536.1 Detailed Description

```
template<typename _Tp>
struct std::_Fwd_list_const_iterator< _Tp >
```

A forward_list::const_iterator.

All the functions are op overloads.

Definition at line 187 of file forward_list.h.

The documentation for this struct was generated from the following file:

- [forward_list.h](#)

5.537 std::_Fwd_list_iterator< _Tp > Struct Template Reference

Public Types

- typedef [_Fwd_list_node](#)< _Tp > **_Node**
- typedef [_Fwd_list_iterator](#)< _Tp > **_Self**
- typedef ptrdiff_t **difference_type**
- typedef [std::forward_iterator_tag](#) **iterator_category**
- typedef _Tp * **pointer**
- typedef _Tp & **reference**
- typedef _Tp **value_type**

Public Member Functions

- [_Fwd_list_iterator](#) ([_Fwd_list_node_base](#) *__n) noexcept
- [_Self _M_next](#) () const noexcept
- bool **operator!=** (const [_Self](#) &__x) const noexcept
- reference **operator*** () const noexcept
- [_Self](#) & **operator++** () noexcept
- [_Self](#) **operator++** (int) noexcept
- pointer **operator->** () const noexcept
- bool **operator==** (const [_Self](#) &__x) const noexcept

Public Attributes

- [_Fwd_list_node_base](#) * **_M_node**

5.537.1 Detailed Description

```
template<typename _Tp>
struct std::_Fwd_list_iterator<_Tp>
```

A forward_list::iterator.

All the functions are op overloads.

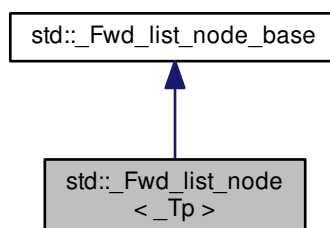
Definition at line 120 of file forward_list.h.

The documentation for this struct was generated from the following file:

- [forward_list.h](#)

5.538 std::_Fwd_list_node<_Tp> Struct Template Reference

Inheritance diagram for std::_Fwd_list_node<_Tp>:



Public Member Functions

- void **_M_reverse_after** () noexcept
- [_Fwd_list_node_base](#) * **_M_transfer_after** ([_Fwd_list_node_base](#) * __begin, [_Fwd_list_node_base](#) * __end) noexcept
- [_Tp](#) * **_M_valptr** () noexcept
- const [_Tp](#) * **_M_valptr** () const noexcept

Public Attributes

- [_Fwd_list_node_base](#) * **_M_next**
- [__gnu_cxx::__aligned_buffer<_Tp>](#) **_M_storage**

5.538.1 Detailed Description

```
template<typename _Tp>
struct std::_Fwd_list_node< _Tp >
```

A helper node class for forward_list. This is just a linked list with uninitialized storage for a data value in each node. There is a sorting utility method.

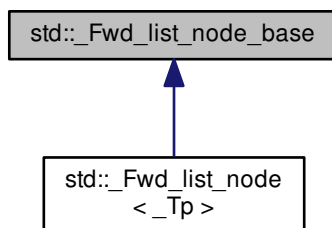
Definition at line 98 of file forward_list.h.

The documentation for this struct was generated from the following file:

- [forward_list.h](#)

5.539 std::_Fwd_list_node_base Struct Reference

Inheritance diagram for std::_Fwd_list_node_base:



Public Member Functions

- void **_M_reverse_after** () noexcept
- [_Fwd_list_node_base](#) * **_M_transfer_after** ([_Fwd_list_node_base](#) * __begin, [_Fwd_list_node_base](#) * __end) noexcept

Public Attributes

- [_Fwd_list_node_base](#) * **_M_next**

5.539.1 Detailed Description

A helper basic node class for forward_list. This is just a linked list with nothing inside it. There are purely list shuffling utility methods here.

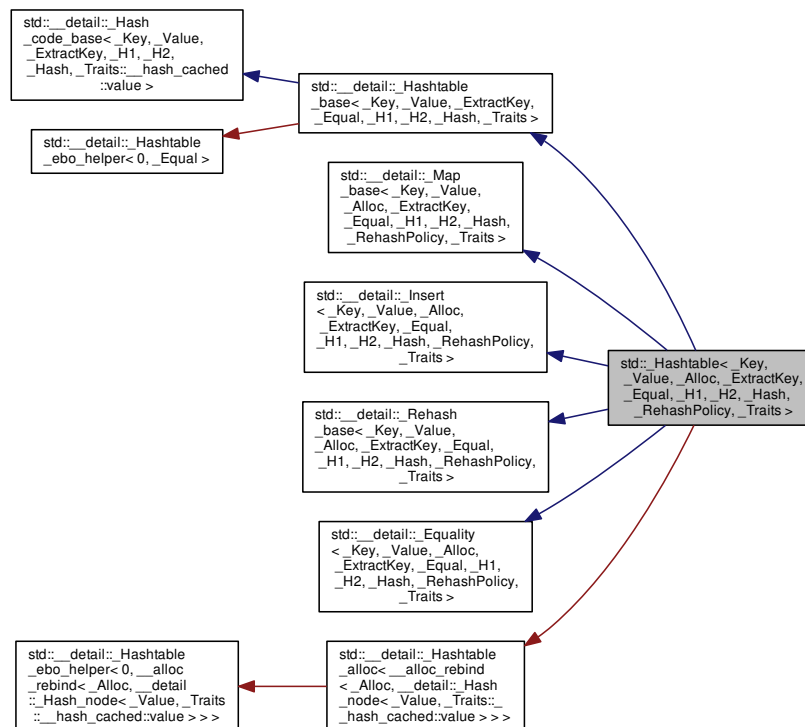
Definition at line 53 of file forward_list.h.

The documentation for this struct was generated from the following file:

- [forward_list.h](#)

5.540 std::_Hashtable< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits > Class Template Reference

Inheritance diagram for std::_Hashtable< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits >:



Public Types

- typedef `_Alloc` **allocator_type**
- using **const_iterator** = typename `__hashtable_base::const_iterator`
- using **const_local_iterator** = typename `__hashtable_base::const_local_iterator`
- typedef `__value_alloc_traits::const_pointer` **const_pointer**
- typedef const value_type & **const_reference**
- using **difference_type** = typename `__hashtable_base::difference_type`
- using **iterator** = typename `__hashtable_base::iterator`
- typedef `__Equal` **key_equal**
- typedef `__Key` **key_type**
- using **local_iterator** = typename `__hashtable_base::local_iterator`
- typedef `__value_alloc_traits::pointer` **pointer**
- typedef value_type & **reference**
- using **size_type** = typename `__hashtable_base::size_type`
- typedef `_Value` **value_type**

Public Member Functions

- **_Hashtable** (size_type __bucket_hint, const _H1 &, const _H2 &, const _Hash &, const _Equal &, const _ExtractKey &, const allocator_type &)
- template<typename _InputIterator >
 _Hashtable (_InputIterator __first, _InputIterator __last, size_type __bucket_hint, const _H1 &, const _H2 &, const _Hash &, const _Equal &, const _ExtractKey &, const allocator_type &)
- **_Hashtable** (const `_Hashtable` &)
- **_Hashtable** (`_Hashtable` &&) noexcept
- **_Hashtable** (const `_Hashtable` &, const allocator_type &)
- **_Hashtable** (`_Hashtable` &&, const allocator_type &)
- **_Hashtable** (const allocator_type & __a)
- **_Hashtable** (size_type __n, const _H1 & __hf=_H1(), const key_equal & __eq=key_equal(), const allocator_type & __a=allocator_type())
- template<typename _InputIterator >
 _Hashtable (_InputIterator __f, _InputIterator __l, size_type __n=0, const _H1 & __hf=_H1(), const key_equal & __eq=key_equal(), const allocator_type & __a=allocator_type())
- **_Hashtable** (initializer_list< value_type > __l, size_type __n=0, const _H1 & __hf=_H1(), const key_equal & __eq=key_equal(), const allocator_type & __a=allocator_type())
- const _RehashPolicy & **__rehash_policy** () const
- void **__rehash_policy** (const _RehashPolicy & __pol)
- template<typename... _Args>
 auto **_M_emplace** (std::true_type, _Args &&... __args) -> pair< iterator, bool >
- template<typename... _Args>
 auto **_M_emplace** (const_iterator __hint, std::false_type, _Args &&... __args) -> iterator
- template<typename _Arg, typename _NodeGenerator >
 auto **_M_insert** (_Arg && __v, const _NodeGenerator & __node_gen, std::true_type) -> pair< iterator, bool >
- template<typename _Arg, typename _NodeGenerator >
 auto **_M_insert** (const_iterator __hint, _Arg && __v, const _NodeGenerator & __node_gen, std::false_type) -> iterator
- iterator **begin** () noexcept
- const_iterator **begin** () const noexcept
- local_iterator **begin** (size_type __n)
- const_local_iterator **begin** (size_type __n) const

- `size_type bucket` (const key_type &__k) const
- `size_type bucket_count` () const noexcept
- `size_type bucket_size` (size_type __n) const
- `const_iterator cbegin` () const noexcept
- `const_local_iterator cbegin` (size_type __n) const
- `const_iterator cend` () const noexcept
- `const_local_iterator cend` (size_type __n) const
- `void clear` () noexcept
- `size_type count` (const key_type &__k) const
- `template<typename... _Args>`
`__ireturn_type emplace` (_Args &&... __args)
- `template<typename... _Args>`
`iterator emplace_hint` (const_iterator __hint, _Args &&... __args)
- `bool empty` () const noexcept
- `iterator end` () noexcept
- `const_iterator end` () const noexcept
- `local_iterator end` (size_type __n)
- `const_local_iterator end` (size_type __n) const
- `std::pair< iterator, iterator > equal_range` (const key_type &__k)
- `std::pair< const_iterator, const_iterator > equal_range` (const key_type &__k) const
- `iterator erase` (const_iterator)
- `iterator erase` (iterator __it)
- `size_type erase` (const key_type &__k)
- `iterator erase` (const_iterator, const_iterator)
- `iterator find` (const key_type &__k)
- `const_iterator find` (const key_type &__k) const
- `allocator_type get_allocator` () const noexcept
- `key_equal key_eq` () const
- `float load_factor` () const noexcept
- `size_type max_bucket_count` () const noexcept
- `size_type max_size` () const noexcept
- `_Hashtable & operator=` (const _Hashtable &__ht)
- `_Hashtable & operator=` (_Hashtable &&__ht) noexcept(__node_alloc_traits::_S_nothrow_move() &&is_nothrow_move_assignable< _H1 >::value &&is_nothrow_move_assignable< _Equal >::value)
- `_Hashtable & operator=` (initializer_list< value_type > __l)
- `void rehash` (size_type __n)
- `size_type size` () const noexcept
- `void swap` (_Hashtable &) noexcept(__is_nothrow_swappable< _H1 >::value &&__is_nothrow_swappable< _Equal >::value)

Protected Member Functions

- `size_type _M_bucket_index` (__node_type *__n) const noexcept
- `size_type _M_bucket_index` (const key_type &__k, __hash_code __c) const
- `template<typename... _Args>`
`std::pair< iterator, bool > _M_emplace` (std::true_type, _Args &&... __args)
- `template<typename... _Args>`
`iterator _M_emplace` (std::false_type __uk, _Args &&... __args)
- `template<typename... _Args>`
`iterator _M_emplace` (const_iterator, std::true_type __uk, _Args &&... __args)

- `template<typename... _Args>`
`iterator _M_emplace (const_iterator, std::false_type, _Args &&... __args)`
- `const _Equal & _M_eq () const`
- `_Equal & _M_eq ()`
- `bool _M_equals (const _Key & __k, __hash_code __c, __node_type * __n) const`
- `size_type _M_erase (std::true_type, const key_type &)`
- `size_type _M_erase (std::false_type, const key_type &)`
- `iterator _M_erase (size_type __bkt, __node_base * __prev_n, __node_type * __n)`
- `__node_base * _M_find_before_node (size_type, const key_type &, __hash_code) const`
- `__node_type * _M_find_node (size_type __bkt, const key_type & __key, __hash_code __c) const`
- `__node_base * _M_get_previous_node (size_type __bkt, __node_base * __n)`
- `template<typename _Arg, typename _NodeGenerator >`
`std::pair< iterator, bool > _M_insert (_Arg &&, const _NodeGenerator &, std::true_type)`
- `template<typename _Arg, typename _NodeGenerator >`
`iterator _M_insert (_Arg && __arg, const _NodeGenerator & __node_gen, std::false_type __uk)`
- `template<typename _Arg, typename _NodeGenerator >`
`iterator _M_insert (const_iterator, _Arg && __arg, const _NodeGenerator & __node_gen, std::true_type __uk)`
- `template<typename _Arg, typename _NodeGenerator >`
`iterator _M_insert (const_iterator, _Arg &&, const _NodeGenerator &, std::false_type)`
- `void _M_insert_bucket_begin (size_type, __node_type *)`
- `iterator _M_insert_multi_node (__node_type * __hint, __hash_code __code, __node_type * __n)`
- `iterator _M_insert_unique_node (size_type __bkt, __hash_code __code, __node_type * __n)`
- `void _M_remove_bucket_begin (size_type __bkt, __node_type * __next_n, size_type __next_bkt)`
- `void _M_swap (_Hashtable_base & __x)`

Private Types

- `using __bucket_alloc_traits = std::allocator_traits< __bucket_alloc_type >`
- `using __bucket_alloc_type = __alloc_rebind< __node_alloc_type, __bucket_type >`
- `using __value_alloc_type = __alloc_rebind< __node_alloc_type, __value_type >`
- `using __value_type = typename __node_type::value_type`

Private Member Functions

- `__bucket_type * _M_allocate_buckets (std::size_t __n)`
- `__node_type * _M_allocate_node (_Args &&... __args)`
- `void _M_deallocate_buckets (__bucket_type *, std::size_t __n)`
- `void _M_deallocate_node (__node_type * __n)`
- `void _M_deallocate_nodes (__node_type * __n)`
- `__node_alloc_type & _M_node_allocator ()`
- `const __node_alloc_type & _M_node_allocator () const`

Friends

- `template<typename _Keya, typename _Valuea, typename _Alloca, typename _ExtractKeya, typename _Equala, typename _H1a, typename _H2a, typename _Hasha, typename _RehashPolicya, typename _Traitsa, bool _Constant_iteratorsa, bool _Unique_keysa>`
`struct __detail::Insert`
- `template<typename _Keya, typename _Valuea, typename _Alloca, typename _ExtractKeya, typename _Equala, typename _H1a, typename _H2a, typename _Hasha, typename _RehashPolicya, typename _Traitsa >`
`struct __detail::Insert_base`
- `template<typename _Keya, typename _Valuea, typename _Alloca, typename _ExtractKeya, typename _Equala, typename _H1a, typename _H2a, typename _Hasha, typename _RehashPolicya, typename _Traitsa, bool _Unique_keysa>`
`struct __detail::Map_base`

5.540.1 Detailed Description

```
template<typename _Key, typename _Value, typename _Alloc, typename _ExtractKey, typename _Equal, typename _H1, typename _H2,
typename _Hash, typename _RehashPolicy, typename _Traits>
class std::_Hashtable< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits >
```

Primary class template _Hashtable.

Template Parameters

<code>_Value</code>	CopyConstructible type.
<code>_Key</code>	CopyConstructible type.
<code>_Alloc</code>	An allocator type ([lib.allocator.requirements]) whose <code>_Alloc::value_type</code> is <code>_Value</code> . As a conforming extension, we allow for <code>_Alloc::value_type != _Value</code> .
<code>_ExtractKey</code>	Function object that takes an object of type <code>_Value</code> and returns a value of type <code>_Key</code> .
<code>_Equal</code>	Function object that takes two objects of type <code>k</code> and returns a bool-like value that is true if the two objects are considered equal.
<code>_H1</code>	The hash function. A unary function object with argument type <code>_Key</code> and result type <code>size_t</code> . Return values should be distributed over the entire range <code>[0, numeric_limits<size_t>:max()]</code> .
<code>_H2</code>	The range-hashing function (in the terminology of Tavori and Dreizin). A binary function object whose argument types and result type are all <code>size_t</code> . Given arguments <code>r</code> and <code>N</code> , the return value is in the range <code>[0, N)</code> .
<code>_Hash</code>	The ranged hash function (Tavori and Dreizin). A binary function whose argument types are <code>_Key</code> and <code>size_t</code> and whose result type is <code>size_t</code> . Given arguments <code>k</code> and <code>N</code> , the return value is in the range <code>[0, N)</code> . Default: <code>hash(k, N) = h2(h1(k), N)</code> . If <code>_Hash</code> is anything other than the default, <code>_H1</code> and <code>_H2</code> are ignored.
<code>_RehashPolicy</code>	Policy class with three members, all of which govern the bucket count. <code>_M_next_bkt(n)</code> returns a bucket count no smaller than <code>n</code> . <code>_M_bkt_for_elements(n)</code> returns a bucket count appropriate for an element count of <code>n</code> . <code>_M_need_rehash(n_bkt, n_elt, n_ins)</code> determines whether, if the current bucket count is <code>n_bkt</code> and the current element count is <code>n_elt</code> , we need to increase the bucket count. If so, returns <code>make_pair(true, n)</code> , where <code>n</code> is the new bucket count. If not, returns <code>make_pair(false, <anything>)</code>
<code>_Traits</code>	Compile-time class with three boolean <code>std::integral_constant</code> members: <code>__cache_hash_code</code> , <code>__constant_iterators</code> , <code>__unique_keys</code> .

Each _Hashtable data structure has:

- `_Bucket[] _M_buckets`
- `_Hash_node_base _M_before_begin`
- `size_type _M_bucket_count`
- `size_type _M_element_count`

with `_Bucket` being `_Hash_node*` and `_Hash_node` containing:

- `_Hash_node* _M_next`

- `Tp _M_value`
- `size_t _M_hash_code` if `cache_hash_code` is true

In terms of Standard containers the hashtable is like the aggregation of:

- `std::forward_list<_Node>` containing the elements
- `std::vector<std::forward_list<_Node>::iterator>` representing the buckets

The non-empty buckets contain the node before the first node in the bucket. This design makes it possible to implement something like `std::forward_list::insert_after` on container insertion and `std::forward_list::erase_after` on container erase calls. `_M_before_begin` is equivalent to `std::forward_list::before_begin`. Empty buckets contain `nullptr`. Note that one of the non-empty buckets contains `&_M_before_begin` which is not a dereferenceable node so the node pointer in a bucket shall never be dereferenced, only its next node can be.

Walking through a bucket's nodes requires a check on the hash code to see if each node is still in the bucket. Such a design assumes a quite efficient hash functor and is one of the reasons it is highly advisable to set `__cache_hash_code` to true.

The container iterators are simply built from nodes. This way incrementing the iterator is perfectly efficient independent of how many empty buckets there are in the container.

On insert we compute the element's hash code and use it to find the bucket index. If the element must be inserted in an empty bucket we add it at the beginning of the singly linked list and make the bucket point to `_M_before_begin`. The bucket that used to point to `_M_before_begin`, if any, is updated to point to its new before begin node.

On erase, the simple iterator design requires using the hash functor to get the index of the bucket to update. For this reason, when `__cache_hash_code` is set to false the hash functor must not throw and this is enforced by a static assertion.

Functionality is implemented by decomposition into base classes, where the derived `_Hashtable` class is used in `_Map_base`, `_Insert`, `_Rehash_base`, and `_Equality` base classes to access the "this" pointer. `_Hashtable_base` is used in the base classes as a non-recursive, fully-completed-type so that detailed nested type information, such as iterator type and node type, can be used. This is similar to the "Curiously Recurring Template Pattern" (CRTP) technique, but uses a reconstructed, not explicitly passed, template pattern.

Base class templates are:

- `__detail::_Hashtable_base`
- `__detail::_Map_base`
- `__detail::_Insert`
- `__detail::_Rehash_base`
- `__detail::_Equality`

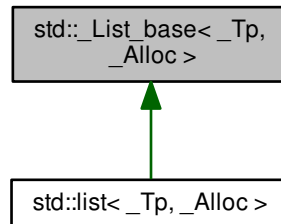
Definition at line 170 of file `bits/hashtable.h`.

The documentation for this class was generated from the following file:

- [bits/hashtable.h](#)

5.541 `std::_List_base<_Tp, _Alloc>` Class Template Reference

Inheritance diagram for `std::_List_base<_Tp, _Alloc>`:



Public Types

- typedef `_Alloc` **allocator_type**

Public Member Functions

- **_List_base** (const `_Node_alloc_type` &__a) noexcept
- **_List_base** (`_List_base` &&__x) noexcept
- **_List_base** (`_List_base` &&__x, `_Node_alloc_type` &&__a)
- void **_M_clear** () noexcept
- `_Node_alloc_type` & **_M_get_Node_allocator** () noexcept
- const `_Node_alloc_type` & **_M_get_Node_allocator** () const noexcept
- void **_M_init** () noexcept
- void **_M_move_nodes** (`_List_base` &&__x)

Protected Types

- typedef `__gnu_cxx::__alloc_traits<_Node_alloc_type>` **_Node_alloc_traits**
- typedef `_Tp_alloc_traits::template rebind<_List_node<_Tp>>::other` **_Node_alloc_type**
- typedef `__gnu_cxx::__alloc_traits<_Tp_alloc_type>` **_Tp_alloc_traits**
- typedef `__gnu_cxx::__alloc_traits<_Alloc>::template rebind<_Tp>::other` **_Tp_alloc_type**

Protected Member Functions

- void **_M_dec_size** (size_t)
- size_t **_M_distance** (const void *, const void *) const
- `_Node_alloc_traits::pointer` **_M_get_node** ()
- size_t **_M_get_size** () const
- void **_M_inc_size** (size_t)
- size_t **_M_node_count** () const
- void **_M_put_node** (typename `_Node_alloc_traits::pointer` __p) noexcept
- void **_M_set_size** (size_t)

Static Protected Member Functions

- static `size_t S_distance` (const [__detail::_List_node_base](#) * __first, const [__detail::_List_node_base](#) * __last)

Protected Attributes

- `_List_impl M_impl`

5.541.1 Detailed Description

```
template<typename _Tp, typename _Alloc>
class std::_List_base< _Tp, _Alloc >
```

See `bits/stl_deque.h`'s `_Deque_base` for an explanation.

Definition at line 300 of file `stl_list.h`.

The documentation for this class was generated from the following files:

- [stl_list.h](#)
- [list.tcc](#)

5.542 `std::_List_const_iterator< _Tp >` Struct Template Reference

Public Types

- typedef const [_List_node](#)< _Tp > **_Node**
- typedef [_List_const_iterator](#)< _Tp > **_Self**
- typedef ptrdiff_t **difference_type**
- typedef [_List_iterator](#)< _Tp > **iterator**
- typedef [std::bidirectional_iterator_tag](#) **iterator_category**
- typedef const _Tp * **pointer**
- typedef const _Tp & **reference**
- typedef _Tp **value_type**

Public Member Functions

- [_List_const_iterator](#) (const [__detail::_List_node_base](#) * __x) noexcept
- [_List_const_iterator](#) (const [iterator](#) & __x) noexcept
- [iterator **M_const_cast**](#) () const noexcept
- bool **operator!=** (const [_Self](#) & __x) const noexcept
- reference **operator*** () const noexcept
- [_Self](#) & **operator++** () noexcept
- [_Self](#) **operator++** (int) noexcept
- [_Self](#) & **operator--** () noexcept
- [_Self](#) **operator--** (int) noexcept
- pointer **operator->** () const noexcept
- bool **operator==** (const [_Self](#) & __x) const noexcept

Public Attributes

- `const __detail::_List_node_base * _M_node`

5.542.1 Detailed Description

```
template<typename _Tp>
struct std::_List_const_iterator<_Tp>
```

A `list::const_iterator`.

All the functions are op overloads.

Definition at line 72 of file `stl_iterator_base_funcs.h`.

The documentation for this struct was generated from the following files:

- [stl_iterator_base_funcs.h](#)
- [stl_list.h](#)

5.543 `std::_List_iterator<_Tp>` Struct Template Reference

Public Types

- `typedef __List_node<_Tp> _Node`
- `typedef __List_iterator<_Tp> _Self`
- `typedef ptrdiff_t difference_type`
- `typedef std::bidirectional_iterator_tag iterator_category`
- `typedef _Tp * pointer`
- `typedef _Tp & reference`
- `typedef _Tp value_type`

Public Member Functions

- `_List_iterator (__detail::_List_node_base *__x) noexcept`
- `_Self _M_const_cast () const noexcept`
- `bool operator!= (const _Self &__x) const noexcept`
- `reference operator* () const noexcept`
- `_Self & operator++ () noexcept`
- `_Self operator++ (int) noexcept`
- `_Self & operator-- () noexcept`
- `_Self operator-- (int) noexcept`
- `pointer operator-> () const noexcept`
- `bool operator== (const _Self &__x) const noexcept`

Public Attributes

- [__detail::_List_node_base](#) * **_M_node**

5.543.1 Detailed Description

```
template<typename _Tp>
struct std::_List_iterator< _Tp >
```

A list::iterator.

All the functions are op overloads.

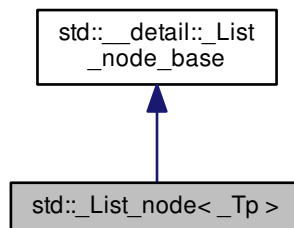
Definition at line 71 of file stl_iterator_base_funcs.h.

The documentation for this struct was generated from the following files:

- [stl_iterator_base_funcs.h](#)
- [stl_list.h](#)

5.544 std::_List_node< _Tp > Struct Template Reference

Inheritance diagram for std::_List_node< _Tp >:



Public Member Functions

- void **_M_hook** (_List_node_base *const __position) noexcept
- void **_M_reverse** () noexcept
- void **_M_transfer** (_List_node_base *const __first, _List_node_base *const __last) noexcept
- void **_M_unhook** () noexcept
- _Tp * **_M_valptr** ()
- _Tp const * **_M_valptr** () const

Static Public Member Functions

- static void **swap** (`_List_node_base &__x`, `_List_node_base &__y`) noexcept

Public Attributes

- `_List_node_base * _M_next`
- `_List_node_base * _M_prev`
- `__gnu_cxx::__aligned_membuf<_Tp > _M_storage`

5.544.1 Detailed Description

```
template<typename _Tp>
struct std::_List_node<_Tp >
```

An actual node in the list.

Definition at line 109 of file `stl_list.h`.

The documentation for this struct was generated from the following file:

- [stl_list.h](#)

5.545 `std::_Maybe_get_result_type<_Functor, typename >` Struct Template Reference

5.545.1 Detailed Description

```
template<typename _Functor, typename = __void_t<>>
struct std::_Maybe_get_result_type<_Functor, typename >
```

If we have found a `result_type`, extract it.

Definition at line 72 of file `functional`.

The documentation for this struct was generated from the following file:

- [functional](#)

5.546 `std::_Maybe_unary_or_binary_function<_Res, _ArgTypes>` Struct Template Reference

5.546.1 Detailed Description

```
template<typename _Res, typename... _ArgTypes>
struct std::_Maybe_unary_or_binary_function<_Res, _ArgTypes>
```

Derives from `unary_function` or `binary_function`, or perhaps nothing, depending on the number of arguments provided. The primary template is the basis case, which derives nothing.

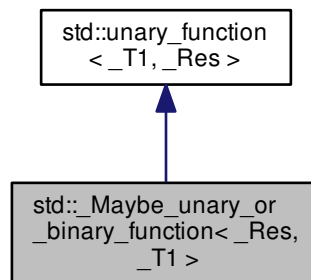
Definition at line 525 of file `functional`.

The documentation for this struct was generated from the following file:

- [functional](#)

5.547 `std::_Maybe_unary_or_binary_function<_Res, _T1>` Struct Template Reference

Inheritance diagram for `std::_Maybe_unary_or_binary_function<_Res, _T1>`:



Public Types

- typedef `_T1` [argument_type](#)
- typedef `_Res` [result_type](#)

5.547.1 Detailed Description

```
template<typename _Res, typename _T1>
struct std::_Maybe_unary_or_binary_function<_Res, _T1>
```

Derives from `unary_function`, as appropriate.

Definition at line 529 of file `functional`.

5.547.2 Member Typedef Documentation

5.547.2.1 typedef _T1 std::unary_function<_T1, _Res>::argument_type [inherited]

argument_type is the type of the argument

Definition at line 108 of file stl_function.h.

5.547.2.2 typedef _Res std::unary_function<_T1, _Res>::result_type [inherited]

result_type is the return type

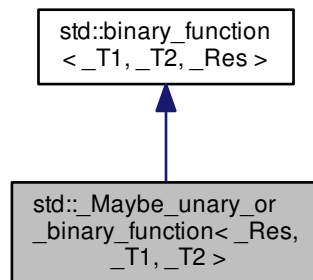
Definition at line 111 of file stl_function.h.

The documentation for this struct was generated from the following file:

- [functional](#)

5.548 std::_Maybe_unary_or_binary_function<_Res, _T1, _T2> Struct Template Reference

Inheritance diagram for std::_Maybe_unary_or_binary_function<_Res, _T1, _T2>:



Public Types

- typedef _T1 [first_argument_type](#)
- typedef _Res [result_type](#)
- typedef _T2 [second_argument_type](#)

5.548.1 Detailed Description

```
template<typename _Res, typename _T1, typename _T2>
struct std::_Maybe_unary_or_binary_function< _Res, _T1, _T2 >
```

Derives from `binary_function`, as appropriate.

Definition at line 534 of file `functional`.

5.548.2 Member Typedef Documentation

5.548.2.1 `typedef _T1 std::binary_function< _T1, _T2, _Res >::first_argument_type` `[inherited]`

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.548.2.2 `typedef _Res std::binary_function< _T1, _T2, _Res >::result_type` `[inherited]`

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.548.2.3 `typedef _T2 std::binary_function< _T1, _T2, _Res >::second_argument_type` `[inherited]`

`second_argument_type` is the type of the second argument

Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [functional](#)

5.549 `std::_Maybe_wrap_member_pointer< _Tp >` Struct Template Reference

Public Types

- `typedef _Tp type`

Static Public Member Functions

- `static constexpr const _Tp & __do_wrap (const _Tp &__x)`
- `static constexpr _Tp && __do_wrap (_Tp &&__x)`

5.549.1 Detailed Description

```
template<typename _Tp>
struct std::_Maybe_wrap_member_pointer<_Tp>
```

Maps member pointers into instances of `_Mem_fn` but leaves all other function objects untouched. Used by `std::bind()`. The primary template handles the non-member-pointer case.

Definition at line 861 of file `functional`.

The documentation for this struct was generated from the following file:

- [functional](#)

5.550 `std::_Maybe_wrap_member_pointer<_Tp_Class::*>` Struct Template Reference

Public Types

- `typedef _Mem_fn<_Tp_Class::*> type`

Static Public Member Functions

- `static constexpr type __do_wrap(_Tp_Class::__pm)`

5.550.1 Detailed Description

```
template<typename _Tp, typename _Class>
struct std::_Maybe_wrap_member_pointer<_Tp_Class::*>
```

Maps member pointers into instances of `_Mem_fn` but leaves all other function objects untouched. Used by `std::bind()`. This partial specialization handles the member pointer case.

Definition at line 880 of file `functional`.

The documentation for this struct was generated from the following file:

- [functional](#)

5.551 `std::_Mu<_Arg, _IsBindExp, _IsPlaceholder >` Class Template Reference

5.551.1 Detailed Description

```
template<typename _Arg, bool _IsBindExp = is_bind_expression<_Arg>::value, bool _IsPlaceholder = (is_placeholder<_Arg>::value > 0)>
class std::_Mu<_Arg, _IsBindExp, _IsPlaceholder >
```

Maps an argument to `bind()` into an actual argument to the bound function object `[func.bind.bind]/10`. Only the first parameter should be specified: the rest are used to determine among the various implementations. Note that, although this class is a function object, it isn't entirely normal because it takes only two parameters regardless of the number of parameters passed to the bind expression. The first parameter is the bound argument and the second parameter is a tuple containing references to the rest of the arguments.

Definition at line 764 of file `functional`.

The documentation for this class was generated from the following file:

- [functional](#)

5.552 `std::_Mu<_Arg, false, false >` Class Template Reference

Public Member Functions

- `template<typename _CVarArg, typename _Tuple >`
`_CVarArg && operator() (_CVarArg &&__arg, _Tuple &) const volatile`

5.552.1 Detailed Description

```
template<typename _Arg>
class std::_Mu<_Arg, false, false >
```

If the argument is just a value, returns a reference to that value. The cv-qualifiers on the reference are determined by the caller. C++11 `[func.bind.bind]` p10 bullet 4.

Definition at line 846 of file `functional`.

The documentation for this class was generated from the following file:

- [functional](#)

5.553 `std::_Mu<_Arg, false, true >` Class Template Reference

Public Member Functions

- `template<typename _Tuple >`
`_Safe_tuple_element_t<(is_placeholder<_Arg >::value-1), _Tuple > && operator() (const volatile _Arg &, _Tuple &__tuple) const volatile`

5.553.1 Detailed Description

```
template<typename _Arg>
class std::_Mu<_Arg, false, true >
```

If the argument is a placeholder for the Nth argument, returns a reference to the Nth argument to the bind function object. C++11 [func.bind.bind] p10 bullet 3.

Definition at line 826 of file functional.

The documentation for this class was generated from the following file:

- [functional](#)

5.554 `std::_Mu<_Arg, true, false >` Class Template Reference

Public Member Functions

- `template<typename _CVArg, typename... _Args>`
`auto operator() (_CVArg &__arg, tuple<_Args... > &__tuple) const volatile-> decltype(__arg(declval<_Args>()...))`

5.554.1 Detailed Description

```
template<typename _Arg>
class std::_Mu<_Arg, true, false >
```

If the argument is a bind expression, we invoke the underlying function object with the same cv-qualifiers as we are given and pass along all of our arguments (unwrapped). C++11 [func.bind.bind] p10 bullet 2.

Definition at line 792 of file functional.

The documentation for this class was generated from the following file:

- [functional](#)

5.555 `std::_Mu<reference_wrapper<_Tp>, false, false >` Class Template Reference

Public Member Functions

- `template<typename _CVRef, typename _Tuple >`
`_Tp & operator() (_CVRef &__arg, _Tuple &) const volatile`

5.555.1 Detailed Description

```
template<typename _Tp>
class std::_Mu< reference_wrapper< _Tp >, false, false >
```

If the argument is `reference_wrapper<_Tp>`, returns the underlying reference. C++11 [func.bind.bind] p10 bullet 1.

Definition at line 772 of file functional.

The documentation for this class was generated from the following file:

- [functional](#)

5.556 std::_Placeholder< _Num > Struct Template Reference

5.556.1 Detailed Description

```
template<int _Num>
struct std::_Placeholder< _Num >
```

The type of placeholder objects defined by libstdc++.

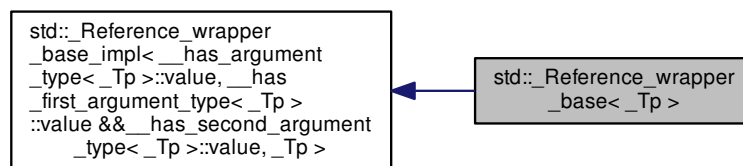
Definition at line 679 of file functional.

The documentation for this struct was generated from the following file:

- [functional](#)

5.557 std::_Reference_wrapper_base< _Tp > Struct Template Reference

Inheritance diagram for `std::_Reference_wrapper_base< _Tp >`:



5.557.1 Detailed Description

```
template<typename _Tp>
struct std::_Reference_wrapper_base<_Tp>
```

Derives from `unary_function` or `binary_function` when it can. Specializations handle all of the easy cases. The primary template determines what to do with a class type, which may derive from both `unary_function` and `binary_function`.

Definition at line 319 of file `functional`.

The documentation for this struct was generated from the following file:

- [functional](#)

5.558 `std::_Reference_wrapper_base_impl<_Unary, _Binary, _Tp>` Struct Template Reference

5.558.1 Detailed Description

```
template<bool _Unary, bool _Binary, typename _Tp>
struct std::_Reference_wrapper_base_impl<_Unary, _Binary, _Tp>
```

Knowing which of `unary_function` and `binary_function` `_Tp` derives from, derives from the same and ensures that `reference_wrapper` will have a weak result type. See cases below.

Definition at line 273 of file `functional`.

The documentation for this struct was generated from the following file:

- [functional](#)

5.559 `std::_Sp_ebo_helper<_Nm, _Tp, false>` Struct Template Reference

Public Member Functions

- `_Sp_ebo_helper` (`const _Tp &__tp`)

Static Public Member Functions

- `static _Tp &_S_get` (`_Sp_ebo_helper &__eboh`)

5.559.1 Detailed Description

```
template<int _Nm, typename _Tp>
struct std::_Sp_ebo_helper< _Nm, _Tp, false >
```

Specialization not using EBO.

Definition at line 417 of file shared_ptr_base.h.

The documentation for this struct was generated from the following file:

- [shared_ptr_base.h](#)

5.560 std::_Sp_ebo_helper< _Nm, _Tp, true > Struct Template Reference

Inherits `_Tp`.

Public Member Functions

- **_Sp_ebo_helper** (const `_Tp` &__tp)

Static Public Member Functions

- static `_Tp` & **_S_get** (`_Sp_ebo_helper` &__eboh)

5.560.1 Detailed Description

```
template<int _Nm, typename _Tp>
struct std::_Sp_ebo_helper< _Nm, _Tp, true >
```

Specialization using EBO.

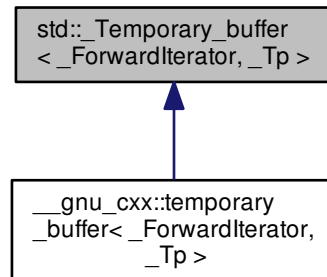
Definition at line 407 of file shared_ptr_base.h.

The documentation for this struct was generated from the following file:

- [shared_ptr_base.h](#)

5.561 `std::_Temporary_buffer<_ForwardIterator, _Tp>` Class Template Reference

Inheritance diagram for `std::_Temporary_buffer<_ForwardIterator, _Tp>`:



Public Types

- typedef pointer **iterator**
- typedef value_type * **pointer**
- typedef ptrdiff_t **size_type**
- typedef _Tp **value_type**

Public Member Functions

- `_Temporary_buffer` (`_ForwardIterator __first`, `_ForwardIterator __last`)
- iterator `begin` ()
- iterator `end` ()
- size_type `requested_size` () const
- size_type `size` () const

Protected Attributes

- pointer **_M_buffer**
- size_type **_M_len**
- size_type **_M_original_len**

5.561.1 Detailed Description

```
template<typename _ForwardIterator, typename _Tp>
class std::_Temporary_buffer<_ForwardIterator, _Tp>
```

This class is used in two places: `stl_algo.h` and `ext/memory`, where it is wrapped as the `temporary_buffer` class. See `temporary_buffer` docs for more notes.

Definition at line 122 of file `stl_tempbuf.h`.

5.561.2 Constructor & Destructor Documentation

5.561.2.1 `template<typename _ForwardIterator, typename _Tp> std::_Temporary_buffer<_ForwardIterator, _Tp>::_Temporary_buffer(_ForwardIterator __first, _ForwardIterator __last)`

Constructs a temporary buffer of a size somewhere between zero and the size of the given range.

Definition at line 244 of file `stl_tempbuf.h`.

References `std::pair<_T1, _T2>::first`, `std::get_temporary_buffer()`, `std::return_temporary_buffer()`, and `std::pair<_T1, _T2>::second`.

Referenced by `std::_Temporary_buffer<_ForwardIterator, _Tp>::end()`.

5.561.3 Member Function Documentation

5.561.3.1 `template<typename _ForwardIterator, typename _Tp> iterator std::_Temporary_buffer<_ForwardIterator, _Tp>::begin() [inline]`

As per Table mumble.

Definition at line 151 of file `stl_tempbuf.h`.

Referenced by `std::__stable_partition_adaptive()`.

5.561.3.2 `template<typename _ForwardIterator, typename _Tp> iterator std::_Temporary_buffer<_ForwardIterator, _Tp>::end() [inline]`

As per Table mumble.

Definition at line 156 of file `stl_tempbuf.h`.

References `std::__addressof()`, `std::_Construct()`, `std::_Destroy()`, `std::_Temporary_buffer<_ForwardIterator, _Tp>::_Temporary_buffer()`, and `std::return_temporary_buffer()`.

5.561.3.3 `template<typename _ForwardIterator, typename _Tp> size_type std::_Temporary_buffer<_ForwardIterator, _Tp>::requested_size() const [inline]`

Returns the size requested by the constructor; may be `>size()`.

Definition at line 146 of file `stl_tempbuf.h`.

Referenced by `std::__stable_partition_adaptive()`.

5.561.3.4 `template<typename _ForwardIterator, typename _Tp> size_type std::_Temporary_buffer<_ForwardIterator, _Tp>::size () const [inline]`

As per Table mumble.

Definition at line 141 of file `stl_tempbuf.h`.

Referenced by `std::__stable_partition_adaptive()`.

The documentation for this class was generated from the following file:

- [stl_tempbuf.h](#)

5.562 `std::_Tuple_impl<_Idx, _Elements>` Struct Template Reference

5.562.1 Detailed Description

```
template<std::size_t _Idx, typename... _Elements>
struct std::_Tuple_impl<_Idx, _Elements>
```

Contains the actual implementation of the `tuple` template, stored as a recursive inheritance hierarchy from the first element (most derived class) to the last (least derived class). The `Idx` parameter gives the 0-based index of the element stored at this point in the hierarchy; we use it to implement a constant-time `get()` operation.

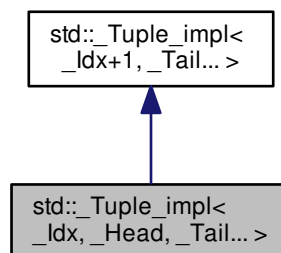
Definition at line 159 of file `tuple`.

The documentation for this struct was generated from the following file:

- [tuple](#)

5.563 `std::_Tuple_impl<_Idx, _Head, _Tail...>` Struct Template Reference

Inheritance diagram for `std::_Tuple_impl<_Idx, _Head, _Tail...>`:



Public Types

- typedef `_Head_base` < `_Idx`, `_Head`, `__empty_not_final` < `_Head` >::value > `_Base`
- typedef `_Tuple_impl` < `_Idx+1`, `_Tail...` > `_Inherited`

Public Member Functions

- constexpr `_Tuple_impl` (const `_Head` & `__head`, const `_Tail` &... `__tail`)
- template<typename `_UHead`, typename... `_UTail`, typename = typename enable_if<sizeof...(`_Tail`) == sizeof...(`_UTail`)>::type>
constexpr `_Tuple_impl` (`_UHead` && `__head`, `_UTail` &&... `__tail`)
- constexpr `_Tuple_impl` (const `_Tuple_impl` &)=default
- constexpr `_Tuple_impl` (`_Tuple_impl` && `__in`) noexcept(`__and` < `is_nothrow_move_constructible` < `_Head` >, `is_nothrow_move_constructible` < `_Inherited` > >::value)
- template<typename... `_UElements`>
constexpr `_Tuple_impl` (const `_Tuple_impl` < `_Idx`, `_UElements...` > & `__in`)
- template<typename `_UHead`, typename... `_UTails`>
constexpr `_Tuple_impl` (`_Tuple_impl` < `_Idx`, `_UHead`, `_UTails...` > && `__in`)
- template<typename `_Alloc` >
`_Tuple_impl` (`allocator_arg_t` `__tag`, const `_Alloc` & `__a`)
- template<typename `_Alloc` >
`_Tuple_impl` (`allocator_arg_t` `__tag`, const `_Alloc` & `__a`, const `_Head` & `__head`, const `_Tail` &... `__tail`)
- template<typename `_Alloc`, typename `_UHead`, typename... `_UTail`, typename = typename enable_if<sizeof...(`_Tail`) == sizeof...(`_U`←`Tail`)>::type>
`_Tuple_impl` (`allocator_arg_t` `__tag`, const `_Alloc` & `__a`, `_UHead` && `__head`, `_UTail` &&... `__tail`)
- template<typename `_Alloc` >
`_Tuple_impl` (`allocator_arg_t` `__tag`, const `_Alloc` & `__a`, const `_Tuple_impl` & `__in`)
- template<typename `_Alloc` >
`_Tuple_impl` (`allocator_arg_t` `__tag`, const `_Alloc` & `__a`, `_Tuple_impl` && `__in`)
- template<typename `_Alloc`, typename... `_UElements`>
`_Tuple_impl` (`allocator_arg_t` `__tag`, const `_Alloc` & `__a`, const `_Tuple_impl` < `_Idx`, `_UElements...` > & `__in`)
- template<typename `_Alloc`, typename `_UHead`, typename... `_UTails`>
`_Tuple_impl` (`allocator_arg_t` `__tag`, const `_Alloc` & `__a`, `_Tuple_impl` < `_Idx`, `_UHead`, `_UTails...` > && `__in`)
- `_Tuple_impl` & `operator=` (const `_Tuple_impl` & `__in`)
- `_Tuple_impl` & `operator=` (`_Tuple_impl` && `__in`) noexcept(`__and` < `is_nothrow_move_assignable` < `_Head` >, `is_nothrow_move_assignable` < `_Inherited` > >::value)
- template<typename... `_UElements`>
`_Tuple_impl` & `operator=` (const `_Tuple_impl` < `_Idx`, `_UElements...` > & `__in`)
- template<typename `_UHead`, typename... `_UTails`>
`_Tuple_impl` & `operator=` (`_Tuple_impl` < `_Idx`, `_UHead`, `_UTails...` > && `__in`)

Static Public Member Functions

- static constexpr `_Head` & `_M_head` (`_Tuple_impl` & `__t`) noexcept
- static constexpr const `_Head` & `_M_head` (const `_Tuple_impl` & `__t`) noexcept
- static constexpr `_Inherited` & `_M_tail` (`_Tuple_impl` & `__t`) noexcept
- static constexpr const `_Inherited` & `_M_tail` (const `_Tuple_impl` & `__t`) noexcept

Protected Member Functions

- void `_M_swap` (`_Tuple_impl` & `__in`) noexcept(`__is_nothrow_swappable` < `_Head` >::value &&noexcept(`_M`←`tail`(`__in`).`_M_swap`(`_M_tail`(`__in`))))

Friends

- `template<std::size_t , typename... >`
`class _Tuple_impl`

5.563.1 Detailed Description

```
template<std::size_t _Idx, typename _Head, typename... _Tail>
struct std::_Tuple_impl<_Idx, _Head, _Tail... >
```

Recursive tuple implementation. Here we store the `Head` element and derive from a `Tuple_impl` containing the remaining elements (which contains the `Tail`).

Definition at line 180 of file `tuple`.

The documentation for this struct was generated from the following file:

- [tuple](#)

5.564 std::_V2::condition_variable_any Class Reference

Public Member Functions

- **condition_variable_any** (const [condition_variable_any](#) &)=delete
- void **notify_all** () noexcept
- void **notify_one** () noexcept
- [condition_variable_any](#) & **operator=** (const [condition_variable_any](#) &)=delete
- `template<typename _Lock >`
void **wait** (_Lock &__lock)
- `template<typename _Lock , typename _Predicate >`
void **wait** (_Lock &__lock, _Predicate __p)
- `template<typename _Lock , typename _Rep , typename _Period >`
[cv_status](#) **wait_for** (_Lock &__lock, const [chrono::duration](#)< _Rep, _Period > &__rtime)
- `template<typename _Lock , typename _Rep , typename _Period , typename _Predicate >`
bool **wait_for** (_Lock &__lock, const [chrono::duration](#)< _Rep, _Period > &__rtime, _Predicate __p)
- `template<typename _Lock , typename _Clock , typename _Duration >`
[cv_status](#) **wait_until** (_Lock &__lock, const [chrono::time_point](#)< _Clock, _Duration > &__atime)
- `template<typename _Lock , typename _Clock , typename _Duration , typename _Predicate >`
bool **wait_until** (_Lock &__lock, const [chrono::time_point](#)< _Clock, _Duration > &__atime, _Predicate __p)

5.564.1 Detailed Description

`condition_variable_any`

Definition at line 187 of file `condition_variable`.

The documentation for this class was generated from the following file:

- [condition_variable](#)

5.565 std::_V2::error_category Class Reference

Public Member Functions

- **error_category** (const [error_category](#) &)=delete
- virtual [error_condition](#) **default_error_condition** (int __i) const noexcept
- virtual bool **equivalent** (int __i, const [error_condition](#) &__cond) const noexcept
- virtual bool **equivalent** (const [error_code](#) &__code, int __i) const noexcept
- virtual [string](#) **message** (int) const =0
- virtual const char * **name** () const noexcept=0
- bool **operator!=** (const [error_category](#) &__other) const noexcept
- bool **operator<** (const [error_category](#) &__other) const noexcept
- [error_category](#) & **operator=** (const [error_category](#) &)=delete
- bool **operator==** (const [error_category](#) &__other) const noexcept

5.565.1 Detailed Description

[error_category](#)

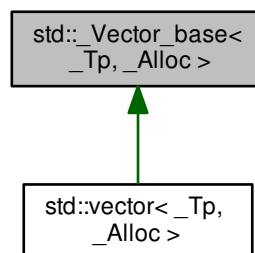
Definition at line 66 of file [system_error](#).

The documentation for this class was generated from the following file:

- [system_error](#)

5.566 std::_Vector_base< _Tp, _Alloc > Struct Template Reference

Inheritance diagram for [std::_Vector_base< _Tp, _Alloc >](#):



Public Types

- typedef [__gnu_cxx::__alloc_traits](#)<_Alloc>::template rebind<_Tp>::other **_Tp_alloc_type**
- typedef _Alloc **allocator_type**
- typedef [__gnu_cxx::__alloc_traits](#)<_Tp_alloc_type>::pointer **pointer**

Public Member Functions

- **_Vector_base** (const allocator_type &__a) noexcept
- **_Vector_base** (size_t __n)
- **_Vector_base** (size_t __n, const allocator_type &__a)
- **_Vector_base** (_Tp_alloc_type &&__a) noexcept
- **_Vector_base** ([_Vector_base](#) &&__x) noexcept
- **_Vector_base** ([_Vector_base](#) &&__x, const allocator_type &__a)
- pointer **_M_allocate** (size_t __n)
- void **_M_deallocate** (pointer __p, size_t __n)
- _Tp_alloc_type & **_M_get_Tp_allocator** () noexcept
- const _Tp_alloc_type & **_M_get_Tp_allocator** () const noexcept
- allocator_type **get_allocator** () const noexcept

Public Attributes

- [_Vector_impl](#) **_M_impl**

5.566.1 Detailed Description

```
template<typename _Tp, typename _Alloc>
struct std::_Vector_base<_Tp, _Alloc>
```

See bits/stl_deque.h's `_Deque_base` for an explanation.

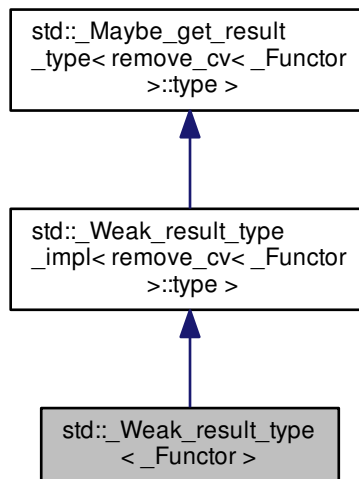
Definition at line 72 of file `stl_vector.h`.

The documentation for this struct was generated from the following file:

- [stl_vector.h](#)

5.567 `std::_Weak_result_type<_Functor>` Struct Template Reference

Inheritance diagram for `std::_Weak_result_type<_Functor>`:



5.567.1 Detailed Description

```
template<typename _Functor>
struct std::_Weak_result_type<_Functor>
```

Strip top-level cv-qualifiers from the function object and let `_Weak_result_type_impl` perform the real work.

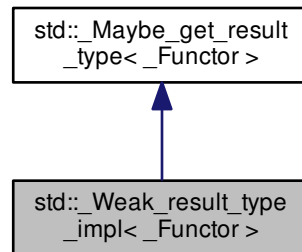
Definition at line 183 of file `functional`.

The documentation for this struct was generated from the following file:

- [functional](#)

5.568 std::_Weak_result_type_impl<_Functor> Struct Template Reference

Inheritance diagram for std::_Weak_result_type_impl<_Functor>:



5.568.1 Detailed Description

```
template<typename _Functor>
struct std::_Weak_result_type_impl<_Functor>
```

Base class for any function object that has a weak result type, as defined in 20.8.2 [func.require] of C++11.

Definition at line 85 of file functional.

The documentation for this struct was generated from the following file:

- [functional](#)

5.569 std::_Weak_result_type_impl<_Res&(_ArgTypes...)> Struct Template Reference

Public Types

- typedef _Res **result_type**

5.569.1 Detailed Description

```
template<typename _Res, typename... _ArgTypes>
struct std::_Weak_result_type_impl<_Res&(_ArgTypes...)>
```

Retrieve the result type for a function reference.

Definition at line 124 of file functional.

The documentation for this struct was generated from the following file:

- [functional](#)

5.570 `std::_Weak_result_type_impl<_Res(*)(_ArgTypes...)>` Struct Template Reference

Public Types

- typedef `_Res` **result_type**

5.570.1 Detailed Description

```
template<typename _Res, typename... _ArgTypes>  
struct std::_Weak_result_type_impl<_Res(*)(_ArgTypes...)>
```

Retrieve the result type for a function pointer.

Definition at line 133 of file functional.

The documentation for this struct was generated from the following file:

- [functional](#)

5.571 `std::_Weak_result_type_impl<_Res(_ArgTypes...)>` Struct Template Reference

Public Types

- typedef `_Res` **result_type**

5.571.1 Detailed Description

```
template<typename _Res, typename... _ArgTypes>  
struct std::_Weak_result_type_impl<_Res(_ArgTypes...)>
```

Retrieve the result type for a function type.

Definition at line 91 of file functional.

The documentation for this struct was generated from the following file:

- [functional](#)

5.572 `std::_Weak_result_type_impl<_Res(_Class::*)(_ArgTypes...) const>` Struct Template Reference

Public Types

- typedef `_Res` **result_type**

5.572.1 Detailed Description

```
template<typename _Res, typename _Class, typename... _ArgTypes>
struct std::_Weak_result_type_impl<_Res(_Class::*)(_ArgTypes...) const >
```

Retrieve result type for a const member function pointer.

Definition at line 151 of file functional.

The documentation for this struct was generated from the following file:

- [functional](#)

5.573 `std::_Weak_result_type_impl<_Res(_Class::*)(_ArgTypes...) const volatile >` Struct Template Reference

Public Types

- typedef `_Res` **result_type**

5.573.1 Detailed Description

```
template<typename _Res, typename _Class, typename... _ArgTypes>
struct std::_Weak_result_type_impl<_Res(_Class::*)(_ArgTypes...) const volatile >
```

Retrieve result type for a const volatile member function pointer.

Definition at line 169 of file functional.

The documentation for this struct was generated from the following file:

- [functional](#)

5.574 `std::_Weak_result_type_impl<_Res(_Class::*)(_ArgTypes...) volatile >` Struct Template Reference

Public Types

- typedef `_Res` **result_type**

5.574.1 Detailed Description

```
template<typename _Res, typename _Class, typename... _ArgTypes>
struct std::_Weak_result_type_impl< _Res(_Class::*)(_ArgTypes...) volatile >
```

Retrieve result type for a volatile member function pointer.

Definition at line 160 of file functional.

The documentation for this struct was generated from the following file:

- [functional](#)

5.575 std::_Weak_result_type_impl< _Res(_Class::*)(_ArgTypes...)> Struct Template Reference

Public Types

- typedef _Res **result_type**

5.575.1 Detailed Description

```
template<typename _Res, typename _Class, typename... _ArgTypes>
struct std::_Weak_result_type_impl< _Res(_Class::*)(_ArgTypes...)>
```

Retrieve result type for a member function pointer.

Definition at line 142 of file functional.

The documentation for this struct was generated from the following file:

- [functional](#)

5.576 std::adopt_lock_t Struct Reference

5.576.1 Detailed Description

Assume the calling thread has already obtained mutex ownership and manage it.

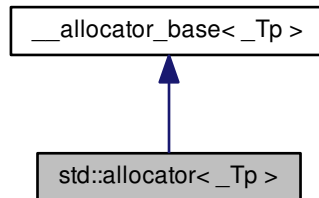
Definition at line 139 of file std_mutex.h.

The documentation for this struct was generated from the following file:

- [std_mutex.h](#)

5.577 std::allocator< _Tp > Class Template Reference

Inheritance diagram for std::allocator< _Tp >:



Public Types

- typedef const _Tp * **const_pointer**
- typedef const _Tp & **const_reference**
- typedef ptrdiff_t **difference_type**
- typedef **true_type** **is_always_equal**
- typedef _Tp * **pointer**
- typedef **true_type** **propagate_on_container_move_assignment**
- typedef _Tp & **reference**
- typedef size_t **size_type**
- typedef _Tp **value_type**

Public Member Functions

- **allocator** (const **allocator** &__a) throw ()
- template<typename _Tp1 >
 allocator (const **allocator**< _Tp1 > &) throw ()
- pointer **address** (reference __x) const noexcept
- const_pointer **address** (const_reference __x) const noexcept
- pointer **allocate** (size_type __n, const void *=0)
- template<typename _Up, typename... _Args>
 void **construct** (_Up *__p, _Args &&... __args)
- void **deallocate** (pointer __p, size_type)
- template<typename _Up >
 void **destroy** (_Up *__p)
- size_type **max_size** () const noexcept

5.577.1 Detailed Description

```
template<typename _Tp>  
class std::allocator<_Tp>
```

The *standard* allocator, as per [20.4].

See [https://gcc.gnu.org/onlinedocs/libstdc++/manual/memory.html#std.util.↵
memory.allocator](https://gcc.gnu.org/onlinedocs/libstdc++/manual/memory.html#std.util.memory.allocator) for further details.

Template Parameters

<code>_Tp</code>	Type of allocated object.
------------------	---------------------------

Definition at line 108 of file `allocator.h`.

The documentation for this class was generated from the following file:

- [allocator.h](#)

5.578 `std::allocator< void >` Class Template Reference

Public Types

- typedef const void * **const_pointer**
- typedef ptrdiff_t **difference_type**
- typedef [true_type](#) **is_always_equal**
- typedef void * **pointer**
- typedef [true_type](#) **propagate_on_container_move_assignment**
- typedef size_t **size_type**
- typedef void **value_type**

Public Member Functions

- template<typename _Up, typename... _Args>
void **construct** (_Up *__p, _Args &&...__args)
- template<typename _Up >
void **destroy** (_Up *__p)

5.578.1 Detailed Description

```
template<>
class std::allocator< void >
```

`allocator<void>` specialization.

Definition at line 68 of file `allocator.h`.

The documentation for this class was generated from the following file:

- [allocator.h](#)

5.579 std::allocator_arg_t Struct Reference

5.579.1 Detailed Description

[allocator.tag]

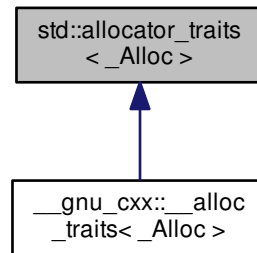
Definition at line 46 of file uses_allocator.h.

The documentation for this struct was generated from the following file:

- uses_allocator.h

5.580 std::allocator_traits<_Alloc> Struct Template Reference

Inheritance diagram for std::allocator_traits<_Alloc>:



Public Types

- template<typename _Alloc, typename _Up>
using **__rebind** = typename _Alloc::template rebind<_Up>::other
- typedef _Alloc **allocator_type**
- using **const_pointer** = __detected_or_t< __ptr_rebind< pointer, const value_type >, __c_pointer, _Alloc >
- using **const_void_pointer** = __detected_or_t< __ptr_rebind< pointer, const void >, __cv_pointer, _Alloc >
- using **difference_type** = __detected_or_t< typename pointer_traits< pointer >::difference_type, __diff_type, __←
_Alloc >
- using **is_always_equal** = __detected_or_t< typename is_empty< _Alloc >::type, __equal, _Alloc >
- using **pointer** = __detected_or_t< value_type *, __pointer, _Alloc >
- using **propagate_on_container_copy_assignment** = __detected_or_t< false_type, __pocca, _Alloc >
- using **propagate_on_container_move_assignment** = __detected_or_t< false_type, __pocma, _Alloc >
- using **propagate_on_container_swap** = __detected_or_t< false_type, __pocs, _Alloc >
- template<typename _Tp>
using **rebind_alloc** = __alloc_rebind< _Alloc, _Tp >
- template<typename _Tp>
using **rebind_traits** = allocator_traits< rebind_alloc< _Tp >>
- using **size_type** = __detected_or_t< typename make_unsigned< difference_type >::type, __size_type, _Alloc >
- typedef _Alloc::value_type **value_type**
- using **void_pointer** = __detected_or_t< __ptr_rebind< pointer, void >, __v_pointer, _Alloc >

Static Public Member Functions

- static [pointer allocate](#) (_Alloc &__a, [size_type](#) __n)
- static [pointer allocate](#) (_Alloc &__a, [size_type](#) __n, [const_void_pointer](#) __hint)
- template<typename _Tp, typename... _Args>
static auto [construct](#) (_Alloc &__a, _Tp *__p, _Args &&... __args) -> decltype(_S_construct(__a, __p, [std::forward](#)<_Args>(__args)...))
- static void [deallocate](#) (_Alloc &__a, [pointer](#) __p, [size_type](#) __n)
- template<typename _Tp >
static void [destroy](#) (_Alloc &__a, _Tp *__p)
- static [size_type max_size](#) (const _Alloc &__a) noexcept
- static _Alloc [select_on_container_copy_construction](#) (const _Alloc &__rhs)

Protected Types

- template<typename _Tp >
using [__c_pointer](#) = typename _Tp::const_pointer
- template<typename _Tp >
using [__cv_pointer](#) = typename _Tp::const_void_pointer
- template<typename _Tp >
using [__diff_type](#) = typename _Tp::difference_type
- template<typename _Tp >
using [__equal](#) = typename _Tp::is_always_equal
- template<typename _Tp >
using [__pocca](#) = typename _Tp::propagate_on_container_copy_assignment
- template<typename _Tp >
using [__pocma](#) = typename _Tp::propagate_on_container_move_assignment
- template<typename _Tp >
using [__pocs](#) = typename _Tp::propagate_on_container_swap
- template<typename _Tp >
using [__pointer](#) = typename _Tp::pointer
- template<typename _Tp >
using [__size_type](#) = typename _Tp::size_type
- template<typename _Tp >
using [__v_pointer](#) = typename _Tp::void_pointer

5.580.1 Detailed Description

```
template<typename _Alloc>
struct std::allocator_traits< _Alloc >
```

Uniform interface to all allocator types.

Definition at line 83 of file bits/alloc_traits.h.

5.580.2 Member Typedef Documentation

5.580.2.1 template<typename _Alloc> typedef _Alloc std::allocator_traits< _Alloc >::allocator_type

The allocator type.

Definition at line 86 of file bits/alloc_traits.h.

5.580.2.2 `template<typename _Alloc> using std::allocator_traits< _Alloc >::const_pointer =
__detected_or_t<__ptr_rebind<pointer, const value_type>, __c_pointer, _Alloc>`

The allocator's const pointer type.

`Alloc::const_pointer` if that type exists, otherwise `pointer_traits<pointer>::rebind<const value_type>`

Definition at line 105 of file `bits/alloc_traits.h`.

5.580.2.3 `template<typename _Alloc> using std::allocator_traits< _Alloc >::const_void_pointer =
__detected_or_t<__ptr_rebind<pointer, const void>, __cv_pointer, _Alloc>`

The allocator's const void pointer type.

`Alloc::const_void_pointer` if that type exists, otherwise `pointer_traits<pointer>::rebind<const void>`

Definition at line 124 of file `bits/alloc_traits.h`.

5.580.2.4 `template<typename _Alloc> using std::allocator_traits< _Alloc >::difference_type = __detected_or_t<typename
pointer_traits<pointer>::difference_type, __diff_type, _Alloc>`

The allocator's difference type.

`Alloc::difference_type` if that type exists, otherwise `pointer_traits<pointer>::difference_type`

Definition at line 134 of file `bits/alloc_traits.h`.

5.580.2.5 `template<typename _Alloc> using std::allocator_traits< _Alloc >::is_always_equal =
__detected_or_t<typename is_empty<_Alloc>::type, __equal, _Alloc>`

Whether all instances of the allocator type compare equal.

`Alloc::is_always_equal` if that type exists, otherwise `is_empty<Alloc>::type`

Definition at line 180 of file `bits/alloc_traits.h`.

5.580.2.6 `template<typename _Alloc> using std::allocator_traits< _Alloc >::pointer = __detected_or_t<value_type*,
__pointer, _Alloc>`

The allocator's pointer type.

`Alloc::pointer` if that type exists, otherwise `value_type*`

Definition at line 95 of file `bits/alloc_traits.h`.

5.580.2.7 `template<typename _Alloc> using std::allocator_traits<_Alloc>::propagate_on_container_copy_↵
assignment = __detected_or_t<false_type, __pocca, _Alloc>`

How the allocator is propagated on copy assignment.

`Alloc::propagate_on_container_copy_assignment` if that type exists, otherwise `false_type`

Definition at line 153 of file `bits/alloc_traits.h`.

5.580.2.8 `template<typename _Alloc> using std::allocator_traits<_Alloc>::propagate_on_container_move_↵
assignment = __detected_or_t<false_type, __pocma, _Alloc>`

How the allocator is propagated on move assignment.

`Alloc::propagate_on_container_move_assignment` if that type exists, otherwise `false_type`

Definition at line 162 of file `bits/alloc_traits.h`.

5.580.2.9 `template<typename _Alloc> using std::allocator_traits<_Alloc>::propagate_on_container_swap =
__detected_or_t<false_type, __pocs, _Alloc>`

How the allocator is propagated on swap.

`Alloc::propagate_on_container_swap` if that type exists, otherwise `false_type`

Definition at line 171 of file `bits/alloc_traits.h`.

5.580.2.10 `template<typename _Alloc> using std::allocator_traits<_Alloc>::size_type = __detected_or_t<typename
make_unsigned<difference_type>::type, __size_type, _Alloc>`

The allocator's size type.

`Alloc::size_type` if that type exists, otherwise `make_unsigned<difference_type>::type`

Definition at line 144 of file `bits/alloc_traits.h`.

5.580.2.11 `template<typename _Alloc> typedef _Alloc::value_type std::allocator_traits<_Alloc>::value_type`

The allocated type.

Definition at line 88 of file `bits/alloc_traits.h`.

5.580.2.12 `template<typename _Alloc> using std::allocator_traits<_Alloc>::void_pointer =
__detected_or_t<__ptr_rebind<pointer, void>, __v_pointer, _Alloc>`

The allocator's void pointer type.

`Alloc::void_pointer` if that type exists, otherwise `pointer_traits<pointer>::rebind<void>`

Definition at line 114 of file `bits/alloc_traits.h`.

5.580.3 Member Function Documentation

5.580.3.1 `template<typename _Alloc> static pointer std::allocator_traits<_Alloc>::allocate (_Alloc & __a, size_type __n
) [inline],[static]`

Allocate memory.

Parameters

$_a$	An allocator.
$_n$	The number of objects to allocate space for.

Calls `a.allocate(n)`

Definition at line 280 of file `bits/alloc_traits.h`.

Referenced by `std::__allocate_guarded()`.

5.580.3.2 `template<typename _Alloc> static pointer std::allocator_traits<_Alloc>::allocate (_Alloc & __a, size_type __n, const_void_pointer __hint) [inline],[static]`

Allocate memory.

Parameters

<code>__a</code>	An allocator.
<code>__n</code>	The number of objects to allocate space for.
<code>__hint</code>	Aid to locality.

Returns

Memory of suitable size and alignment for *n* objects of type `value_type`

Returns `a.allocate(n, hint)` if that expression is well-formed, otherwise returns `a.allocate(n)`

Definition at line 295 of file `bits/alloc_traits.h`.

5.580.3.3 `template<typename _Alloc> template<typename _Tp, typename... _Args> static auto std::allocator_traits<_Alloc>::construct (_Alloc & __a, _Tp * __p, _Args &&... __args) -> decltype(S_construct(__a, __p, std::forward<_Args>(__args)...)) [inline],[static]`

Construct an object of type `_Tp`.

Parameters

<code>__a</code>	An allocator.
<code>__p</code>	Pointer to memory of suitable size and alignment for <code>Tp</code>
<code>__args</code>	Constructor arguments.

Calls `__a.construct(__p, std::forward<Args>(__args)...) if that expression is well-formed, otherwise uses placement-new to construct an object of type _Tp at location __p from the arguments __args...`

Definition at line 322 of file `bits/alloc_traits.h`.

5.580.3.4 `template<typename _Alloc> static void std::allocator_traits<_Alloc>::deallocate (_Alloc & __a, pointer __p, size_type __n) [inline], [static]`

Deallocate memory.

Parameters

<code>__a</code>	An allocator.
<code>__p</code>	Pointer to the memory to deallocate.
<code>__n</code>	The number of objects space was allocated for.

Calls `a.deallocate(p, n)`

Definition at line 307 of file `bits/alloc_traits.h`.

Referenced by `std::__allocated_ptr<_Alloc>::~~__allocated_ptr()`.

5.580.3.5 `template<typename _Alloc> template<typename _Tp> static void std::allocator_traits<_Alloc>::destroy (_Alloc & __a, _Tp* __p) [inline], [static]`

Destroy an object of type `_Tp`.

Parameters

<code>__a</code>	An allocator.
<code>__p</code>	Pointer to the object to destroy

Calls `__a.destroy(__p)` if that expression is well-formed, otherwise calls `__p->~_Tp()`

Definition at line 335 of file `bits/alloc_traits.h`.

5.580.3.6 `template<typename _Alloc> static size_type std::allocator_traits<_Alloc>::max_size (const _Alloc & __a) [inline], [static], [noexcept]`

The maximum supported allocation size.

Parameters

<code>__a</code>	An allocator.
------------------	---------------

Returns

`__a.max_size()` or `numeric_limits<size_type>::max()`

Returns `__a.max_size()` if that expression is well-formed, otherwise returns `numeric_limits<size_type>::max()`

Definition at line 346 of file `bits/alloc_traits.h`.

5.580.3.7 `template<typename _Alloc> static _Alloc std::allocator_traits<_Alloc>::select_on_container_copy_construction (const _Alloc & __rhs) [inline], [static]`

Obtain an allocator to use when copying a container.

Parameters

<code>__rhs</code>	An allocator.
--------------------	---------------

Returns

`__rhs.select_on_container_copy_construction()` or `__rhs`

Returns `__rhs.select_on_container_copy_construction()` if that expression is well-formed, otherwise returns `__rhs`

Definition at line 358 of file `bits/alloc_traits.h`.

The documentation for this struct was generated from the following file:

- [bits/alloc_traits.h](#)

5.581 std::allocator_traits< allocator< _Tp > > Struct Template Reference**Public Types**

- using `allocator_type` = `allocator< _Tp >`
- using `const_pointer` = `const _Tp *`
- using `const_void_pointer` = `const void *`
- using `difference_type` = `std::ptrdiff_t`
- using `is_always_equal` = `true_type`
- using `pointer` = `_Tp *`
- using `propagate_on_container_copy_assignment` = `false_type`
- using `propagate_on_container_move_assignment` = `true_type`
- using `propagate_on_container_swap` = `false_type`
- `template<typename _Up >`
using `rebind_alloc` = `allocator< _Up >`
- `template<typename _Up >`
using `rebind_traits` = `allocator_traits< allocator< _Up >>`
- using `size_type` = `std::size_t`
- using `value_type` = `_Tp`
- using `void_pointer` = `void *`

Static Public Member Functions

- static [pointer allocate](#) ([allocator_type](#) &__a, [size_type](#) __n)
- static [pointer allocate](#) ([allocator_type](#) &__a, [size_type](#) __n, [const_void_pointer](#) __hint)
- template<typename _Up, typename... _Args>
static void [construct](#) ([allocator_type](#) &__a, _Up *__p, _Args &&...__args)
- static void [deallocate](#) ([allocator_type](#) &__a, [pointer](#) __p, [size_type](#) __n)
- template<typename _Up >
static void [destroy](#) ([allocator_type](#) &__a, _Up *__p)
- static [size_type max_size](#) (const [allocator_type](#) &__a) noexcept
- static [allocator_type select_on_container_copy_construction](#) (const [allocator_type](#) &__rhs)

5.581.1 Detailed Description

```
template<typename _Tp>
struct std::allocator_traits< allocator< _Tp > >
```

Partial specialization for std::allocator.

Definition at line 364 of file bits/alloc_traits.h.

5.581.2 Member Typedef Documentation

5.581.2.1 `template<typename _Tp > using std::allocator_traits< allocator< _Tp > >::allocator_type = allocator<_Tp>`

The allocator type.

Definition at line 367 of file bits/alloc_traits.h.

5.581.2.2 `template<typename _Tp > using std::allocator_traits< allocator< _Tp > >::const_pointer = const _Tp*`

The allocator's const pointer type.

Definition at line 375 of file bits/alloc_traits.h.

5.581.2.3 `template<typename _Tp > using std::allocator_traits< allocator< _Tp > >::const_void_pointer = const void*`

The allocator's const void pointer type.

Definition at line 381 of file bits/alloc_traits.h.

5.581.2.4 `template<typename _Tp > using std::allocator_traits< allocator< _Tp > >::difference_type = std::ptrdiff_t`

The allocator's difference type.

Definition at line 384 of file bits/alloc_traits.h.

5.581.2.5 `template<typename _Tp> using std::allocator_traits< allocator< _Tp> >::is_always_equal = true_type`

Whether all instances of the allocator type compare equal.

Definition at line 399 of file `bits/alloc_traits.h`.

5.581.2.6 `template<typename _Tp> using std::allocator_traits< allocator< _Tp> >::pointer = _Tp*`

The allocator's pointer type.

Definition at line 372 of file `bits/alloc_traits.h`.

**5.581.2.7 `template<typename _Tp> using std::allocator_traits< allocator< _Tp> >::propagate_on_container_↵
copy_assignment = false_type`**

How the allocator is propagated on copy assignment.

Definition at line 390 of file `bits/alloc_traits.h`.

**5.581.2.8 `template<typename _Tp> using std::allocator_traits< allocator< _Tp> >::propagate_on_container_↵
move_assignment = true_type`**

How the allocator is propagated on move assignment.

Definition at line 393 of file `bits/alloc_traits.h`.

**5.581.2.9 `template<typename _Tp> using std::allocator_traits< allocator< _Tp> >::propagate_on_container_swap
= false_type`**

How the allocator is propagated on swap.

Definition at line 396 of file `bits/alloc_traits.h`.

5.581.2.10 `template<typename _Tp> using std::allocator_traits< allocator< _Tp> >::size_type = std::size_t`

The allocator's size type.

Definition at line 387 of file `bits/alloc_traits.h`.

5.581.2.11 `template<typename _Tp> using std::allocator_traits< allocator< _Tp> >::value_type = _Tp`

The allocated type.

Definition at line 369 of file `bits/alloc_traits.h`.

5.581.2.12 `template<typename _Tp> using std::allocator_traits< allocator< _Tp> >::void_pointer = void*`

The allocator's void pointer type.

Definition at line 378 of file `bits/alloc_traits.h`.

5.581.3 Member Function Documentation

**5.581.3.1 `template<typename _Tp> static pointer std::allocator_traits< allocator< _Tp> >::allocate (allocator_type
&_a, size_type _n) [inline],[static]`**

Allocate memory.

Parameters

$_a$	An allocator.
$_n$	The number of objects to allocate space for.

Calls `a.allocate(n)`

Definition at line 415 of file `bits/alloc_traits.h`.

5.581.3.2 `template<typename _Tp> static pointer std::allocator_traits< allocator< _Tp > >::allocate (allocator_type &__a, size_type __n, const_void_pointer __hint) [inline],[static]`

Allocate memory.

Parameters

<code>__a</code>	An allocator.
<code>__n</code>	The number of objects to allocate space for.
<code>__hint</code>	Aid to locality.

Returns

Memory of suitable size and alignment for *n* objects of type `value_type`

Returns `a.allocate(n, hint)`

Definition at line 429 of file `bits/alloc_traits.h`.

5.581.3.3 `template<typename _Tp> template<typename _Up, typename... _Args> static void std::allocator_traits< allocator< _Tp > >::construct (allocator_type &__a, _Up * __p, _Args &&... __args) [inline],[static]`

Construct an object of type `_Up`.

Parameters

<code>__a</code>	An allocator.
<code>__p</code>	Pointer to memory of suitable size and alignment for <code>Tp</code>
<code>__args</code>	Constructor arguments.

Calls `__a.construct(__p, std::forward<Args>(__args)...)`

Definition at line 454 of file `bits/alloc_traits.h`.

5.581.3.4 `template<typename _Tp> static void std::allocator_traits< allocator< _Tp > >::deallocate (allocator_type & __a, pointer __p, size_type __n) [inline],[static]`

Deallocate memory.

Parameters

<code>__a</code>	An allocator.
<code>__p</code>	Pointer to the memory to deallocate.
<code>__n</code>	The number of objects space was allocated for.

Calls `a.deallocate(p, n)`

Definition at line 441 of file `bits/alloc_traits.h`.

5.581.3.5 `template<typename _Tp> template<typename _Up> static void std::allocator_traits< allocator< _Tp > >::destroy (allocator_type & __a, _Up * __p) [inline],[static]`

Destroy an object of type `_Up`.

Parameters

<code>__a</code>	An allocator.
<code>__p</code>	Pointer to the object to destroy

Calls `__a.destroy(__p)`.

Definition at line 466 of file `bits/alloc_traits.h`.

5.581.3.6 `template<typename _Tp> static size_type std::allocator_traits< allocator< _Tp > >::max_size (const allocator_type & __a) [inline],[static],[noexcept]`

The maximum supported allocation size.

Parameters

<code>__a</code>	An allocator.
------------------	---------------

Returns

`__a.max_size()`

Definition at line 475 of file `bits/alloc_traits.h`.

```
5.581.3.7 template<typename _Tp> static allocator_type std::allocator_traits< allocator< _Tp>
>::select_on_container_copy_construction ( const allocator_type & __rhs ) [inline],[static]
```

Obtain an allocator to use when copying a container.

Parameters

<code>__rhs</code>	An allocator.
--------------------	---------------

Returns

`__rhs`

Definition at line 484 of file `bits/alloc_traits.h`.

The documentation for this struct was generated from the following file:

- [bits/alloc_traits.h](#)

5.582 std::array<_Tp, _Nm> Struct Template Reference

Public Types

- typedef ::__array_traits<_Tp, _Nm> **_AT_Type**
- typedef const value_type * **const_iterator**
- typedef const value_type * **const_pointer**
- typedef const value_type & **const_reference**
- typedef [std::reverse_iterator](#)< const_iterator > **const_reverse_iterator**
- typedef std::ptrdiff_t **difference_type**
- typedef value_type * **iterator**
- typedef value_type * **pointer**
- typedef value_type & **reference**
- typedef [std::reverse_iterator](#)< iterator > **reverse_iterator**
- typedef std::size_t **size_type**
- typedef _Tp **value_type**

Public Member Functions

- reference **at** (size_type __n)
- constexpr const_reference **at** (size_type __n) const
- reference **back** () noexcept
- constexpr const_reference **back** () const noexcept
- iterator **begin** () noexcept
- const_iterator **begin** () const noexcept
- const_iterator **cbegin** () const noexcept
- const_iterator **cend** () const noexcept
- [const_reverse_iterator](#) **crbegin** () const noexcept

- [const_reverse_iterator](#) **crend** () const noexcept
- pointer **data** () noexcept
- const_pointer **data** () const noexcept
- constexpr bool **empty** () const noexcept
- iterator **end** () noexcept
- const_iterator **end** () const noexcept
- void **fill** (const value_type &__u)
- reference **front** () noexcept
- constexpr const_reference **front** () const noexcept
- constexpr size_type **max_size** () const noexcept
- reference **operator[]** (size_type __n) noexcept
- constexpr const_reference **operator[]** (size_type __n) const noexcept
- [reverse_iterator](#) **rbegin** () noexcept
- [const_reverse_iterator](#) **rbegin** () const noexcept
- [reverse_iterator](#) **rend** () noexcept
- [const_reverse_iterator](#) **rend** () const noexcept
- constexpr size_type **size** () const noexcept
- void **swap** ([array](#) &__other) noexcept(__is_nothrow_swappable<_Tp>::value)

Public Attributes

- `_AT_Type::Type` **_M_elems**

5.582.1 Detailed Description

```
template<typename _Tp, std::size_t _Nm>
struct std::array<_Tp, _Nm >
```

A standard container for storing a fixed size sequence of elements.

Meets the requirements of a [container](#), a [reversible container](#), and a [sequence](#).

Sets support random access iterators.

Template Parameters

<i>Tp</i>	Type of element. Required to be a complete type.
<i>N</i>	Number of elements.

Definition at line 90 of file array.

The documentation for this struct was generated from the following file:

- [array](#)

5.583 std::atomic< _Tp > Struct Template Reference

Public Member Functions

- **atomic** (const [atomic](#) &)=delete
- constexpr **atomic** (_Tp __i) noexcept
- bool **compare_exchange_strong** (_Tp &__e, _Tp __i, [memory_order](#) __s, [memory_order](#) __f) noexcept
- bool **compare_exchange_strong** (_Tp &__e, _Tp __i, [memory_order](#) __s, [memory_order](#) __f) volatile noexcept
- bool **compare_exchange_strong** (_Tp &__e, _Tp __i, [memory_order](#) __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_strong** (_Tp &__e, _Tp __i, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- bool **compare_exchange_weak** (_Tp &__e, _Tp __i, [memory_order](#) __s, [memory_order](#) __f) noexcept
- bool **compare_exchange_weak** (_Tp &__e, _Tp __i, [memory_order](#) __s, [memory_order](#) __f) volatile noexcept
- bool **compare_exchange_weak** (_Tp &__e, _Tp __i, [memory_order](#) __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_weak** (_Tp &__e, _Tp __i, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- _Tp **exchange** (_Tp __i, [memory_order](#) __m=memory_order_seq_cst) noexcept
- _Tp **exchange** (_Tp __i, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- _Tp **load** ([memory_order](#) __m=memory_order_seq_cst) const noexcept
- _Tp **load** ([memory_order](#) __m=memory_order_seq_cst) const volatile noexcept
- **operator _Tp** () const noexcept
- **operator _Tp** () const volatile noexcept
- [atomic](#) & **operator=** (const [atomic](#) &)=delete
- [atomic](#) & **operator=** (const [atomic](#) &) volatile=delete
- _Tp **operator=** (_Tp __i) noexcept
- _Tp **operator=** (_Tp __i) volatile noexcept
- void **store** (_Tp __i, [memory_order](#) __m=memory_order_seq_cst) noexcept
- void **store** (_Tp __i, [memory_order](#) __m=memory_order_seq_cst) volatile noexcept

5.583.1 Detailed Description

```
template<typename _Tp>
struct std::atomic< _Tp >
```

Generic atomic type, primary class template.

Template Parameters

_Tp	Type to be made atomic, must be trivially copyable.
---------------------	---

Definition at line 53 of file atomic.

The documentation for this struct was generated from the following file:

- [atomic](#)

5.584 `std::atomic<_Tp*>` Struct Template Reference

Public Types

- typedef `__atomic_base<_Tp*>` `__base_type`
- typedef `_Tp*` `__pointer_type`

Public Member Functions

- **atomic** (const `atomic` &)=delete
- constexpr **atomic** (`__pointer_type` __p) noexcept
- bool **compare_exchange_strong** (`__pointer_type` &__p1, `__pointer_type` __p2, `memory_order` __m1, `memory_order` __m2) noexcept
- bool **compare_exchange_strong** (`__pointer_type` &__p1, `__pointer_type` __p2, `memory_order` __m1, `memory_order` __m2) volatile noexcept
- bool **compare_exchange_strong** (`__pointer_type` &__p1, `__pointer_type` __p2, `memory_order` __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_strong** (`__pointer_type` &__p1, `__pointer_type` __p2, `memory_order` __m=memory_order_seq_cst) volatile noexcept
- bool **compare_exchange_weak** (`__pointer_type` &__p1, `__pointer_type` __p2, `memory_order` __m1, `memory_order` __m2) noexcept
- bool **compare_exchange_weak** (`__pointer_type` &__p1, `__pointer_type` __p2, `memory_order` __m1, `memory_order` __m2) volatile noexcept
- bool **compare_exchange_weak** (`__pointer_type` &__p1, `__pointer_type` __p2, `memory_order` __m=memory_order_seq_cst) noexcept
- bool **compare_exchange_weak** (`__pointer_type` &__p1, `__pointer_type` __p2, `memory_order` __m=memory_order_seq_cst) volatile noexcept
- `__pointer_type` **exchange** (`__pointer_type` __p, `memory_order` __m=memory_order_seq_cst) noexcept
- `__pointer_type` **exchange** (`__pointer_type` __p, `memory_order` __m=memory_order_seq_cst) volatile noexcept
- `__pointer_type` **fetch_add** (ptrdiff_t __d, `memory_order` __m=memory_order_seq_cst) noexcept
- `__pointer_type` **fetch_add** (ptrdiff_t __d, `memory_order` __m=memory_order_seq_cst) volatile noexcept
- `__pointer_type` **fetch_sub** (ptrdiff_t __d, `memory_order` __m=memory_order_seq_cst) noexcept
- `__pointer_type` **fetch_sub** (ptrdiff_t __d, `memory_order` __m=memory_order_seq_cst) volatile noexcept
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- `__pointer_type` **load** (`memory_order` __m=memory_order_seq_cst) const noexcept
- `__pointer_type` **load** (`memory_order` __m=memory_order_seq_cst) const volatile noexcept
- **operator __pointer_type** () const noexcept
- **operator __pointer_type** () const volatile noexcept
- `__pointer_type` **operator++** (int) noexcept
- `__pointer_type` **operator++** (int) volatile noexcept
- `__pointer_type` **operator++** () noexcept
- `__pointer_type` **operator++** () volatile noexcept
- `__pointer_type` **operator+=** (ptrdiff_t __d) noexcept
- `__pointer_type` **operator+=** (ptrdiff_t __d) volatile noexcept
- `__pointer_type` **operator--** (int) noexcept
- `__pointer_type` **operator--** (int) volatile noexcept
- `__pointer_type` **operator--** () noexcept
- `__pointer_type` **operator--** () volatile noexcept
- `__pointer_type` **operator-=** (ptrdiff_t __d) noexcept

- `__pointer_type operator=` (`ptrdiff_t __d`) volatile noexcept
- `atomic & operator=` (const `atomic &`)=delete
- `atomic & operator=` (const `atomic &`) volatile=delete
- `__pointer_type operator=` (`__pointer_type __p`) noexcept
- `__pointer_type operator=` (`__pointer_type __p`) volatile noexcept
- void `store` (`__pointer_type __p`, `memory_order __m=memory_order_seq_cst`) noexcept
- void `store` (`__pointer_type __p`, `memory_order __m=memory_order_seq_cst`) volatile noexcept

Public Attributes

- `__base_type _M_b`

5.584.1 Detailed Description

```
template<typename _Tp>
struct std::atomic< _Tp * >
```

Partial specialization for pointer types.

Definition at line 320 of file atomic.

The documentation for this struct was generated from the following file:

- `atomic`

5.585 std::atomic< bool > Struct Template Reference

Public Member Functions

- `atomic` (const `atomic &`)=delete
- constexpr `atomic` (bool `__i`) noexcept
- bool `compare_exchange_strong` (bool &`__i1`, bool `__i2`, `memory_order __m1`, `memory_order __m2`) noexcept
- bool `compare_exchange_strong` (bool &`__i1`, bool `__i2`, `memory_order __m1`, `memory_order __m2`) volatile noexcept
- bool `compare_exchange_strong` (bool &`__i1`, bool `__i2`, `memory_order __m=memory_order_seq_cst`) noexcept
- bool `compare_exchange_strong` (bool &`__i1`, bool `__i2`, `memory_order __m=memory_order_seq_cst`) volatile noexcept
- bool `compare_exchange_weak` (bool &`__i1`, bool `__i2`, `memory_order __m1`, `memory_order __m2`) noexcept
- bool `compare_exchange_weak` (bool &`__i1`, bool `__i2`, `memory_order __m1`, `memory_order __m2`) volatile noexcept
- bool `compare_exchange_weak` (bool &`__i1`, bool `__i2`, `memory_order __m=memory_order_seq_cst`) noexcept
- bool `compare_exchange_weak` (bool &`__i1`, bool `__i2`, `memory_order __m=memory_order_seq_cst`) volatile noexcept
- bool `exchange` (bool `__i`, `memory_order __m=memory_order_seq_cst`) noexcept
- bool `exchange` (bool `__i`, `memory_order __m=memory_order_seq_cst`) volatile noexcept
- bool `is_lock_free` () const noexcept
- bool `is_lock_free` () const volatile noexcept

- `bool load (memory_order __m=memory_order_seq_cst) const noexcept`
- `bool load (memory_order __m=memory_order_seq_cst) const volatile noexcept`
- `operator bool () const noexcept`
- `operator bool () const volatile noexcept`
- `atomic & operator= (const atomic &)=delete`
- `atomic & operator= (const atomic &) volatile=delete`
- `bool operator= (bool __i) noexcept`
- `bool operator= (bool __i) volatile noexcept`
- `void store (bool __i, memory_order __m=memory_order_seq_cst) noexcept`
- `void store (bool __i, memory_order __m=memory_order_seq_cst) volatile noexcept`

5.585.1 Detailed Description

```
template<>
struct std::atomic< bool >
```

```
atomic<bool>
```

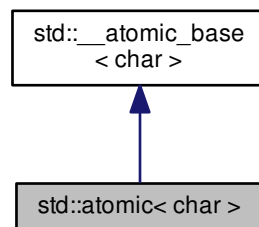
Definition at line 58 of file atomic.

The documentation for this struct was generated from the following file:

- [atomic](#)

5.586 std::atomic< char > Struct Template Reference

Inheritance diagram for `std::atomic< char >`:



Public Types

- `typedef __atomic_base< char > __base_type`
- `typedef char __integral_type`

Public Member Functions

- **atomic** (const [atomic](#) &)=delete
- constexpr **atomic** (__integral_type __i) noexcept
- **__attribute__** ((__always_inline__)) void store(__int_type __i
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- **operator** __int_type () const noexcept
- **operator** __int_type () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept
- __int_type **operator&=** (__int_type __i) volatile noexcept
- __int_type **operator++** (int) noexcept
- __int_type **operator++** (int) volatile noexcept
- __int_type **operator++** () noexcept
- __int_type **operator++** () volatile noexcept
- __int_type **operator+=** (__int_type __i) noexcept
- __int_type **operator+=** (__int_type __i) volatile noexcept
- __int_type **operator--** (int) noexcept
- __int_type **operator--** (int) volatile noexcept
- __int_type **operator--** () noexcept
- __int_type **operator--** () volatile noexcept
- __int_type **operator-=** (__int_type __i) noexcept
- __int_type **operator-=** (__int_type __i) volatile noexcept
- [atomic](#) & **operator=** (const [atomic](#) &)=delete
- [atomic](#) & **operator=** (const [atomic](#) &) volatile=delete
- __int_type **operator^=** (__int_type __i) noexcept
- __int_type **operator^=** (__int_type __i) volatile noexcept
- __int_type **operator|=** (__int_type __i) noexcept
- __int_type **operator|=** (__int_type __i) volatile noexcept

5.586.1 Detailed Description

```
template<>
struct std::atomic< char >
```

Explicit specialization for char.

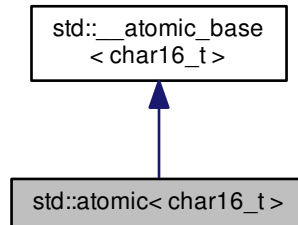
Definition at line 510 of file atomic.

The documentation for this struct was generated from the following file:

- [atomic](#)

5.587 `std::atomic< char16_t >` Struct Template Reference

Inheritance diagram for `std::atomic< char16_t >`:



Public Types

- typedef `__atomic_base< char16_t >` `__base_type`
- typedef `char16_t` `__integral_type`

Public Member Functions

- **atomic** (const `atomic` &)=delete
- constexpr **atomic** (`__integral_type` __i) noexcept
- **__attribute__** ((__always_inline__)) void store(`__int_type` __i
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- **operator __int_type** () const noexcept
- **operator __int_type** () const volatile noexcept
- `__int_type` **operator&=** (`__int_type` __i) noexcept
- `__int_type` **operator&=** (`__int_type` __i) volatile noexcept
- `__int_type` **operator++** (int) noexcept
- `__int_type` **operator++** (int) volatile noexcept
- `__int_type` **operator++** () noexcept
- `__int_type` **operator++** () volatile noexcept
- `__int_type` **operator+=** (`__int_type` __i) noexcept
- `__int_type` **operator+=** (`__int_type` __i) volatile noexcept
- `__int_type` **operator--** (int) noexcept
- `__int_type` **operator--** (int) volatile noexcept
- `__int_type` **operator--** () noexcept
- `__int_type` **operator--** () volatile noexcept
- `__int_type` **operator-=** (`__int_type` __i) noexcept
- `__int_type` **operator-=** (`__int_type` __i) volatile noexcept
- `atomic` & **operator=** (const `atomic` &)=delete
- `atomic` & **operator=** (const `atomic` &) volatile=delete
- `__int_type` **operator^=** (`__int_type` __i) noexcept
- `__int_type` **operator^=** (`__int_type` __i) volatile noexcept
- `__int_type` **operator|=** (`__int_type` __i) noexcept
- `__int_type` **operator|=** (`__int_type` __i) volatile noexcept

5.587.1 Detailed Description

```
template<>
struct std::atomic< char16_t >
```

Explicit specialization for char16_t.

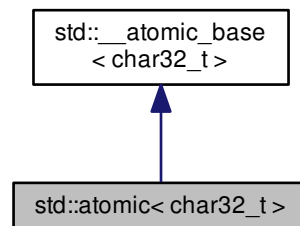
Definition at line 738 of file atomic.

The documentation for this struct was generated from the following file:

- [atomic](#)

5.588 std::atomic< char32_t > Struct Template Reference

Inheritance diagram for std::atomic< char32_t >:



Public Types

- typedef [__atomic_base](#)< char32_t > **__base_type**
- typedef char32_t **__integral_type**

Public Member Functions

- **atomic** (const [atomic](#) &)=delete
- constexpr **atomic** (__integral_type __i) noexcept
- **__attribute__** ((__always_inline__)) void store(__int_type __i
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- **operator __int_type** () const noexcept
- **operator __int_type** () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept

- `__int_type operator&= (__int_type __i) volatile noexcept`
- `__int_type operator++ (int) noexcept`
- `__int_type operator++ (int) volatile noexcept`
- `__int_type operator++ () noexcept`
- `__int_type operator++ () volatile noexcept`
- `__int_type operator+= (__int_type __i) noexcept`
- `__int_type operator+= (__int_type __i) volatile noexcept`
- `__int_type operator-- (int) noexcept`
- `__int_type operator-- (int) volatile noexcept`
- `__int_type operator-- () noexcept`
- `__int_type operator-- () volatile noexcept`
- `__int_type operator-= (__int_type __i) noexcept`
- `__int_type operator-= (__int_type __i) volatile noexcept`
- `atomic & operator= (const atomic &)=delete`
- `atomic & operator= (const atomic &) volatile=delete`
- `__int_type operator^= (__int_type __i) noexcept`
- `__int_type operator^= (__int_type __i) volatile noexcept`
- `__int_type operator|= (__int_type __i) noexcept`
- `__int_type operator|= (__int_type __i) volatile noexcept`

5.588.1 Detailed Description

```
template<>
struct std::atomic< char32_t >
```

Explicit specialization for `char32_t`.

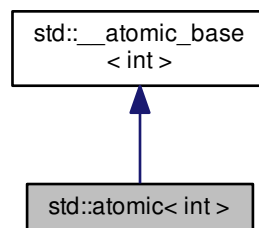
Definition at line 757 of file `atomic`.

The documentation for this struct was generated from the following file:

- [atomic](#)

5.589 std::atomic< int > Struct Template Reference

Inheritance diagram for `std::atomic< int >`:



Public Types

- typedef [__atomic_base](#)< int > **__base_type**
- typedef int **__integral_type**

Public Member Functions

- **atomic** (const [atomic](#) &)=delete
- constexpr **atomic** (__integral_type __i) noexcept
- **__attribute__** ((__always_inline__)) void store(__int_type __i
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- **operator** __int_type () const noexcept
- **operator** __int_type () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept
- __int_type **operator&=** (__int_type __i) volatile noexcept
- __int_type **operator++** (int) noexcept
- __int_type **operator++** (int) volatile noexcept
- __int_type **operator++** () noexcept
- __int_type **operator++** () volatile noexcept
- __int_type **operator+=** (__int_type __i) noexcept
- __int_type **operator+=** (__int_type __i) volatile noexcept
- __int_type **operator--** (int) noexcept
- __int_type **operator--** (int) volatile noexcept
- __int_type **operator--** () noexcept
- __int_type **operator--** () volatile noexcept
- __int_type **operator-=** (__int_type __i) noexcept
- __int_type **operator-=** (__int_type __i) volatile noexcept
- [atomic](#) & **operator=** (const [atomic](#) &)=delete
- [atomic](#) & **operator=** (const [atomic](#) &) volatile=delete
- __int_type **operator^=** (__int_type __i) noexcept
- __int_type **operator^=** (__int_type __i) volatile noexcept
- __int_type **operator|=** (__int_type __i) noexcept
- __int_type **operator|=** (__int_type __i) volatile noexcept

5.589.1 Detailed Description

```
template<>
struct std::atomic< int >
```

Explicit specialization for int.

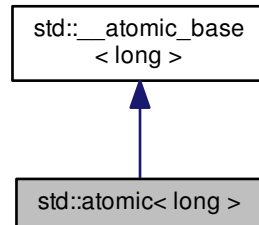
Definition at line 605 of file atomic.

The documentation for this struct was generated from the following file:

- [atomic](#)

5.590 `std::atomic< long >` Struct Template Reference

Inheritance diagram for `std::atomic< long >`:



Public Types

- typedef `__atomic_base< long >` `__base_type`
- typedef `long` `__integral_type`

Public Member Functions

- **atomic** (const `atomic` &)=delete
- constexpr **atomic** (`__integral_type` __i) noexcept
- **__attribute__** ((__always_inline__)) void store(`__int_type` __i
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- **operator __int_type** () const noexcept
- **operator __int_type** () const volatile noexcept
- `__int_type` **operator&=** (`__int_type` __i) noexcept
- `__int_type` **operator&=** (`__int_type` __i) volatile noexcept
- `__int_type` **operator++** (int) noexcept
- `__int_type` **operator++** (int) volatile noexcept
- `__int_type` **operator++** () noexcept
- `__int_type` **operator++** () volatile noexcept
- `__int_type` **operator+=** (`__int_type` __i) noexcept
- `__int_type` **operator+=** (`__int_type` __i) volatile noexcept
- `__int_type` **operator--** (int) noexcept
- `__int_type` **operator--** (int) volatile noexcept
- `__int_type` **operator--** () noexcept
- `__int_type` **operator--** () volatile noexcept
- `__int_type` **operator-=** (`__int_type` __i) noexcept
- `__int_type` **operator-=** (`__int_type` __i) volatile noexcept
- `atomic` & **operator=** (const `atomic` &)=delete
- `atomic` & **operator=** (const `atomic` &) volatile=delete
- `__int_type` **operator^=** (`__int_type` __i) noexcept
- `__int_type` **operator^=** (`__int_type` __i) volatile noexcept
- `__int_type` **operator|=** (`__int_type` __i) noexcept
- `__int_type` **operator|=** (`__int_type` __i) volatile noexcept

5.590.1 Detailed Description

```
template<>
struct std::atomic< long >
```

Explicit specialization for long.

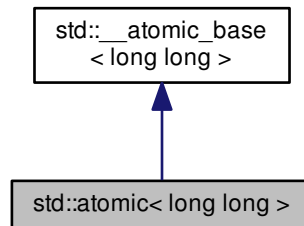
Definition at line 643 of file atomic.

The documentation for this struct was generated from the following file:

- [atomic](#)

5.591 std::atomic< long long > Struct Template Reference

Inheritance diagram for std::atomic< long long >:



Public Types

- typedef [__atomic_base](#)< long long > **__base_type**
- typedef long long **__integral_type**

Public Member Functions

- **atomic** (const [atomic](#) &)=delete
- constexpr **atomic** (__integral_type __i) noexcept
- **__attribute__** ((__always_inline__)) void store(__int_type __i
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- **operator __int_type** () const noexcept
- **operator __int_type** () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept

- `__int_type operator&= (__int_type __i) volatile noexcept`
- `__int_type operator++ (int) noexcept`
- `__int_type operator++ (int) volatile noexcept`
- `__int_type operator++ () noexcept`
- `__int_type operator++ () volatile noexcept`
- `__int_type operator+= (__int_type __i) noexcept`
- `__int_type operator+= (__int_type __i) volatile noexcept`
- `__int_type operator-- (int) noexcept`
- `__int_type operator-- (int) volatile noexcept`
- `__int_type operator-- () noexcept`
- `__int_type operator-- () volatile noexcept`
- `__int_type operator-= (__int_type __i) noexcept`
- `__int_type operator-= (__int_type __i) volatile noexcept`
- `atomic & operator= (const atomic &)=delete`
- `atomic & operator= (const atomic &) volatile=delete`
- `__int_type operator^= (__int_type __i) noexcept`
- `__int_type operator^= (__int_type __i) volatile noexcept`
- `__int_type operator|= (__int_type __i) noexcept`
- `__int_type operator|= (__int_type __i) volatile noexcept`

5.591.1 Detailed Description

```
template<>
struct std::atomic< long long >
```

Explicit specialization for long long.

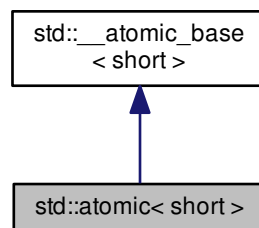
Definition at line 681 of file atomic.

The documentation for this struct was generated from the following file:

- [atomic](#)

5.592 std::atomic< short > Struct Template Reference

Inheritance diagram for std::atomic< short >:



Public Types

- typedef [__atomic_base](#)< short > **__base_type**
- typedef short **__integral_type**

Public Member Functions

- **atomic** (const [atomic](#) &)=delete
- constexpr **atomic** ([__integral_type](#) __i) noexcept
- **__attribute__** ((__always_inline__)) void store([__int_type](#) __i
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- **operator** [__int_type](#) () const noexcept
- **operator** [__int_type](#) () const volatile noexcept
- [__int_type](#) **operator&=** ([__int_type](#) __i) noexcept
- [__int_type](#) **operator&=** ([__int_type](#) __i) volatile noexcept
- [__int_type](#) **operator++** (int) noexcept
- [__int_type](#) **operator++** (int) volatile noexcept
- [__int_type](#) **operator++** () noexcept
- [__int_type](#) **operator++** () volatile noexcept
- [__int_type](#) **operator+=** ([__int_type](#) __i) noexcept
- [__int_type](#) **operator+=** ([__int_type](#) __i) volatile noexcept
- [__int_type](#) **operator--** (int) noexcept
- [__int_type](#) **operator--** (int) volatile noexcept
- [__int_type](#) **operator--** () noexcept
- [__int_type](#) **operator--** () volatile noexcept
- [__int_type](#) **operator-=** ([__int_type](#) __i) noexcept
- [__int_type](#) **operator-=** ([__int_type](#) __i) volatile noexcept
- [atomic](#) & **operator=** (const [atomic](#) &)=delete
- [atomic](#) & **operator=** (const [atomic](#) &) volatile=delete
- [__int_type](#) **operator^=** ([__int_type](#) __i) noexcept
- [__int_type](#) **operator^=** ([__int_type](#) __i) volatile noexcept
- [__int_type](#) **operator|**= ([__int_type](#) __i) noexcept
- [__int_type](#) **operator|**= ([__int_type](#) __i) volatile noexcept

5.592.1 Detailed Description

```
template<>
struct std::atomic< short >
```

Explicit specialization for short.

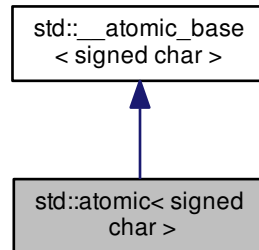
Definition at line 567 of file atomic.

The documentation for this struct was generated from the following file:

- [atomic](#)

5.593 std::atomic< signed char > Struct Template Reference

Inheritance diagram for std::atomic< signed char >:



Public Types

- typedef `__atomic_base< signed char > __base_type`
- typedef `signed char __integral_type`

Public Member Functions

- **atomic** (const **atomic** &)=delete
- constexpr **atomic** (__integral_type __i) noexcept
- **__attribute__** ((__always_inline__)) void store(__int_type __i
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- **operator __int_type** () const noexcept
- **operator __int_type** () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept
- __int_type **operator&=** (__int_type __i) volatile noexcept
- __int_type **operator++** (int) noexcept
- __int_type **operator++** (int) volatile noexcept
- __int_type **operator++** () noexcept
- __int_type **operator++** () volatile noexcept
- __int_type **operator+=** (__int_type __i) noexcept
- __int_type **operator+=** (__int_type __i) volatile noexcept
- __int_type **operator--** (int) noexcept
- __int_type **operator--** (int) volatile noexcept
- __int_type **operator--** () noexcept
- __int_type **operator--** () volatile noexcept
- __int_type **operator-=** (__int_type __i) noexcept
- __int_type **operator-=** (__int_type __i) volatile noexcept
- **atomic** & **operator=** (const **atomic** &)=delete
- **atomic** & **operator=** (const **atomic** &) volatile=delete
- __int_type **operator^=** (__int_type __i) noexcept
- __int_type **operator^=** (__int_type __i) volatile noexcept
- __int_type **operator|=** (__int_type __i) noexcept
- __int_type **operator|=** (__int_type __i) volatile noexcept

5.593.1 Detailed Description

```
template<>
struct std::atomic< signed char >
```

Explicit specialization for signed char.

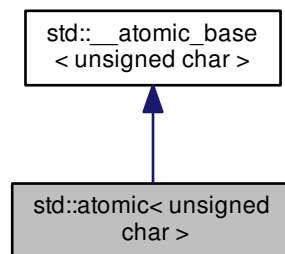
Definition at line 529 of file atomic.

The documentation for this struct was generated from the following file:

- [atomic](#)

5.594 std::atomic< unsigned char > Struct Template Reference

Inheritance diagram for std::atomic< unsigned char >:



Public Types

- typedef [__atomic_base](#)< unsigned char > **__base_type**
- typedef unsigned char **__integral_type**

Public Member Functions

- **atomic** (const [atomic](#) &)=delete
- constexpr **atomic** (__integral_type __i) noexcept
- **__attribute__** ((__always_inline__)) void store(__int_type __i
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- **operator __int_type** () const noexcept
- **operator __int_type** () const volatile noexcept

- `__int_type operator&= (__int_type __i) noexcept`
- `__int_type operator&= (__int_type __i) volatile noexcept`
- `__int_type operator++ (int) noexcept`
- `__int_type operator++ (int) volatile noexcept`
- `__int_type operator++ () noexcept`
- `__int_type operator++ () volatile noexcept`
- `__int_type operator+= (__int_type __i) noexcept`
- `__int_type operator+= (__int_type __i) volatile noexcept`
- `__int_type operator-- (int) noexcept`
- `__int_type operator-- (int) volatile noexcept`
- `__int_type operator-- () noexcept`
- `__int_type operator-- () volatile noexcept`
- `__int_type operator-= (__int_type __i) noexcept`
- `__int_type operator-= (__int_type __i) volatile noexcept`
- `atomic & operator= (const atomic &)=delete`
- `atomic & operator= (const atomic &) volatile=delete`
- `__int_type operator^= (__int_type __i) noexcept`
- `__int_type operator^= (__int_type __i) volatile noexcept`
- `__int_type operator|= (__int_type __i) noexcept`
- `__int_type operator|= (__int_type __i) volatile noexcept`

5.594.1 Detailed Description

`template<>`

`struct std::atomic< unsigned char >`

Explicit specialization for unsigned char.

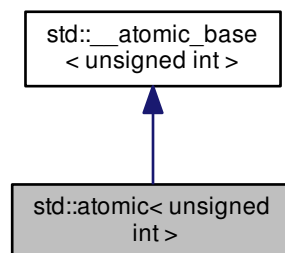
Definition at line 548 of file atomic.

The documentation for this struct was generated from the following file:

- [atomic](#)

5.595 `std::atomic< unsigned int >` Struct Template Reference

Inheritance diagram for `std::atomic< unsigned int >`:



Public Types

- typedef [__atomic_base](#)< unsigned int > **__base_type**
- typedef unsigned int **__integral_type**

Public Member Functions

- **atomic** (const [atomic](#) &)=delete
- constexpr **atomic** (__integral_type __i) noexcept
- **__attribute__** ((__always_inline__)) void store(__int_type __i
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- **operator __int_type** () const noexcept
- **operator __int_type** () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept
- __int_type **operator&=** (__int_type __i) volatile noexcept
- __int_type **operator++** (int) noexcept
- __int_type **operator++** (int) volatile noexcept
- __int_type **operator++** () noexcept
- __int_type **operator++** () volatile noexcept
- __int_type **operator+=** (__int_type __i) noexcept
- __int_type **operator+=** (__int_type __i) volatile noexcept
- __int_type **operator--** (int) noexcept
- __int_type **operator--** (int) volatile noexcept
- __int_type **operator--** () noexcept
- __int_type **operator--** () volatile noexcept
- __int_type **operator-=** (__int_type __i) noexcept
- __int_type **operator-=** (__int_type __i) volatile noexcept
- [atomic](#) & **operator=** (const [atomic](#) &)=delete
- [atomic](#) & **operator=** (const [atomic](#) &) volatile=delete
- __int_type **operator^=** (__int_type __i) noexcept
- __int_type **operator^=** (__int_type __i) volatile noexcept
- __int_type **operator|=** (__int_type __i) noexcept
- __int_type **operator|=** (__int_type __i) volatile noexcept

5.595.1 Detailed Description

template<>

struct std::atomic< unsigned int >

Explicit specialization for unsigned int.

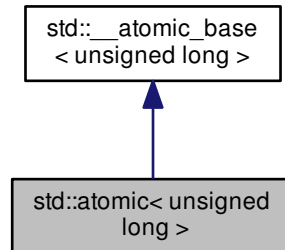
Definition at line 624 of file atomic.

The documentation for this struct was generated from the following file:

- [atomic](#)

5.596 std::atomic< unsigned long > Struct Template Reference

Inheritance diagram for std::atomic< unsigned long >:



Public Types

- typedef `__atomic_base< unsigned long > __base_type`
- typedef `unsigned long __integral_type`

Public Member Functions

- **atomic** (const **atomic** &)=delete
- constexpr **atomic** (__integral_type __i) noexcept
- **__attribute__** ((__always_inline__)) void store(__int_type __i
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- **operator __int_type** () const noexcept
- **operator __int_type** () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept
- __int_type **operator&=** (__int_type __i) volatile noexcept
- __int_type **operator++** (int) noexcept
- __int_type **operator++** (int) volatile noexcept
- __int_type **operator++** () noexcept
- __int_type **operator++** () volatile noexcept
- __int_type **operator+=** (__int_type __i) noexcept
- __int_type **operator+=** (__int_type __i) volatile noexcept
- __int_type **operator--** (int) noexcept
- __int_type **operator--** (int) volatile noexcept
- __int_type **operator--** () noexcept
- __int_type **operator--** () volatile noexcept
- __int_type **operator-=** (__int_type __i) noexcept
- __int_type **operator-=** (__int_type __i) volatile noexcept
- **atomic** & **operator=** (const **atomic** &)=delete
- **atomic** & **operator=** (const **atomic** &) volatile=delete
- __int_type **operator^=** (__int_type __i) noexcept
- __int_type **operator^=** (__int_type __i) volatile noexcept
- __int_type **operator|=** (__int_type __i) noexcept
- __int_type **operator|=** (__int_type __i) volatile noexcept

5.596.1 Detailed Description

```
template<>
struct std::atomic< unsigned long >
```

Explicit specialization for unsigned long.

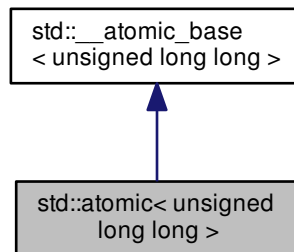
Definition at line 662 of file atomic.

The documentation for this struct was generated from the following file:

- [atomic](#)

5.597 `std::atomic< unsigned long long >` Struct Template Reference

Inheritance diagram for `std::atomic< unsigned long long >`:



Public Types

- typedef `__atomic_base< unsigned long long >` `__base_type`
- typedef `unsigned long long` `__integral_type`

Public Member Functions

- **atomic** (const `atomic` &)=delete
- constexpr **atomic** (`__integral_type __i`) noexcept
- **__attribute__** ((__always_inline__)) void store(`__int_type __i`
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- **operator __int_type** () const noexcept
- **operator __int_type** () const volatile noexcept

- `__int_type operator&= (__int_type __i) noexcept`
- `__int_type operator&= (__int_type __i) volatile noexcept`
- `__int_type operator++ (int) noexcept`
- `__int_type operator++ (int) volatile noexcept`
- `__int_type operator++ () noexcept`
- `__int_type operator++ () volatile noexcept`
- `__int_type operator+= (__int_type __i) noexcept`
- `__int_type operator+= (__int_type __i) volatile noexcept`
- `__int_type operator-- (int) noexcept`
- `__int_type operator-- (int) volatile noexcept`
- `__int_type operator-- () noexcept`
- `__int_type operator-- () volatile noexcept`
- `__int_type operator-= (__int_type __i) noexcept`
- `__int_type operator-= (__int_type __i) volatile noexcept`
- `atomic & operator= (const atomic &)=delete`
- `atomic & operator= (const atomic &) volatile=delete`
- `__int_type operator^= (__int_type __i) noexcept`
- `__int_type operator^= (__int_type __i) volatile noexcept`
- `__int_type operator|= (__int_type __i) noexcept`
- `__int_type operator|= (__int_type __i) volatile noexcept`

5.597.1 Detailed Description

`template<>`

`struct std::atomic< unsigned long long >`

Explicit specialization for unsigned long long.

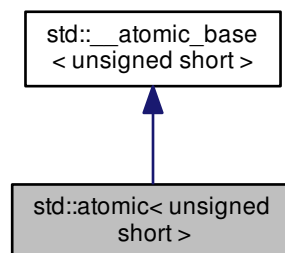
Definition at line 700 of file atomic.

The documentation for this struct was generated from the following file:

- [atomic](#)

5.598 `std::atomic< unsigned short >` Struct Template Reference

Inheritance diagram for `std::atomic< unsigned short >`:



Public Types

- typedef [__atomic_base](#)< unsigned short > **__base_type**
- typedef unsigned short **__integral_type**

Public Member Functions

- **atomic** (const [atomic](#) &)=delete
- constexpr **atomic** (__integral_type __i) noexcept
- **__attribute__** ((__always_inline__)) void store(__int_type __i
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- **operator __int_type** () const noexcept
- **operator __int_type** () const volatile noexcept
- __int_type **operator&=** (__int_type __i) noexcept
- __int_type **operator&=** (__int_type __i) volatile noexcept
- __int_type **operator++** (int) noexcept
- __int_type **operator++** (int) volatile noexcept
- __int_type **operator++** () noexcept
- __int_type **operator++** () volatile noexcept
- __int_type **operator+=** (__int_type __i) noexcept
- __int_type **operator+=** (__int_type __i) volatile noexcept
- __int_type **operator--** (int) noexcept
- __int_type **operator--** (int) volatile noexcept
- __int_type **operator--** () noexcept
- __int_type **operator--** () volatile noexcept
- __int_type **operator-=** (__int_type __i) noexcept
- __int_type **operator-=** (__int_type __i) volatile noexcept
- [atomic](#) & **operator=** (const [atomic](#) &)=delete
- [atomic](#) & **operator=** (const [atomic](#) &) volatile=delete
- __int_type **operator^=** (__int_type __i) noexcept
- __int_type **operator^=** (__int_type __i) volatile noexcept
- __int_type **operator|=** (__int_type __i) noexcept
- __int_type **operator|=** (__int_type __i) volatile noexcept

5.598.1 Detailed Description

template<>

struct std::atomic< unsigned short >

Explicit specialization for unsigned short.

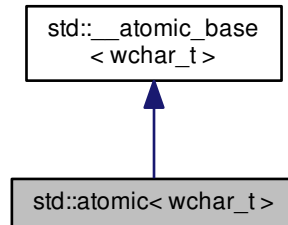
Definition at line 586 of file atomic.

The documentation for this struct was generated from the following file:

- [atomic](#)

5.599 `std::atomic< wchar_t >` Struct Template Reference

Inheritance diagram for `std::atomic< wchar_t >`:



Public Types

- typedef `__atomic_base< wchar_t > __base_type`
- typedef `wchar_t __integral_type`

Public Member Functions

- **atomic** (const `atomic` &)=delete
- constexpr **atomic** (`__integral_type __i`) noexcept
- **__attribute__** ((__always_inline__)) void store(`__int_type __i`
- bool **is_lock_free** () const noexcept
- bool **is_lock_free** () const volatile noexcept
- **operator __int_type** () const noexcept
- **operator __int_type** () const volatile noexcept
- `__int_type` **operator&=** (`__int_type __i`) noexcept
- `__int_type` **operator&=** (`__int_type __i`) volatile noexcept
- `__int_type` **operator++** (int) noexcept
- `__int_type` **operator++** (int) volatile noexcept
- `__int_type` **operator++** () noexcept
- `__int_type` **operator++** () volatile noexcept
- `__int_type` **operator+=** (`__int_type __i`) noexcept
- `__int_type` **operator+=** (`__int_type __i`) volatile noexcept
- `__int_type` **operator--** (int) noexcept
- `__int_type` **operator--** (int) volatile noexcept
- `__int_type` **operator--** () noexcept
- `__int_type` **operator--** () volatile noexcept
- `__int_type` **operator-=** (`__int_type __i`) noexcept
- `__int_type` **operator-=** (`__int_type __i`) volatile noexcept
- `atomic` & **operator=** (const `atomic` &)=delete
- `atomic` & **operator=** (const `atomic` &) volatile=delete
- `__int_type` **operator^=** (`__int_type __i`) noexcept
- `__int_type` **operator^=** (`__int_type __i`) volatile noexcept
- `__int_type` **operator|=** (`__int_type __i`) noexcept
- `__int_type` **operator|=** (`__int_type __i`) volatile noexcept

5.599.1 Detailed Description

```
template<>
struct std::atomic< wchar_t >
```

Explicit specialization for wchar_t.

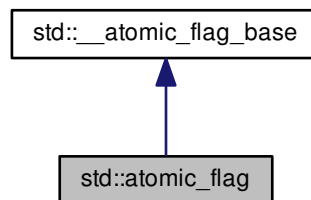
Definition at line 719 of file atomic.

The documentation for this struct was generated from the following file:

- [atomic](#)

5.600 std::atomic_flag Struct Reference

Inheritance diagram for std::atomic_flag:



Public Member Functions

- **atomic_flag** (const [atomic_flag](#) &)=delete
- constexpr **atomic_flag** (bool __i) noexcept
- [atomic_flag](#) & **operator=** (const [atomic_flag](#) &)=delete
- [atomic_flag](#) & **operator=** (const [atomic_flag](#) &) volatile=delete

Public Attributes

- __atomic_flag_data_type **_M_i**

5.600.1 Detailed Description

atomic_flag

Definition at line 160 of file atomic_base.h.

The documentation for this struct was generated from the following file:

- [atomic_base.h](#)

5.601 std::auto_ptr<_Tp> Class Template Reference

Public Types

- typedef _Tp [element_type](#)

Public Member Functions

- [auto_ptr](#) ([element_type](#) *__p=0) throw ()
- [auto_ptr](#) ([auto_ptr](#) &__a) throw ()
- template<typename _Tp1 >
[auto_ptr](#) ([auto_ptr](#)<_Tp1> &__a) throw ()
- [auto_ptr](#) ([auto_ptr_ref](#)<[element_type](#)> __ref) throw ()
- ~[auto_ptr](#) ()
- [element_type](#) * [get](#) () const throw ()
- template<typename _Tp1 >
operator [auto_ptr](#)<_Tp1> () throw ()
- template<typename _Tp1 >
operator [auto_ptr_ref](#)<_Tp1> () throw ()
- [element_type](#) & [operator*](#) () const throw ()
- [element_type](#) * [operator->](#) () const throw ()
- [auto_ptr](#) & [operator=](#) ([auto_ptr](#) &__a) throw ()
- template<typename _Tp1 >
[auto_ptr](#) & [operator=](#) ([auto_ptr](#)<_Tp1> &__a) throw ()
- [auto_ptr](#) & [operator=](#) ([auto_ptr_ref](#)<[element_type](#)> __ref) throw ()
- [element_type](#) * [release](#) () throw ()
- void [reset](#) ([element_type](#) *__p=0) throw ()

5.601.1 Detailed Description

```
template<typename _Tp>  
class std::auto_ptr<_Tp>
```

A simple smart pointer providing strict ownership semantics.

The Standard says:

An `auto_ptr` owns the object it holds a pointer to. Copying an `auto_ptr` copies the pointer and transfers ownership to the destination. If more than one `auto_ptr` owns the same object at the same time the behavior of the program is undefined.

The uses of `auto_ptr` include providing temporary exception-safety for dynamically allocated memory, passing ownership of dynamically allocated memory to a function, and returning dynamically allocated memory from a function. `auto_ptr` does not meet the CopyConstructible requirements for Standard Library `container` elements and thus instantiating a Standard Library container with an `auto_ptr` results in undefined behavior.

Quoted from [20.4.5]/3.

Good examples of what can and cannot be done with `auto_ptr` can be found in the libstdc++ testsuite.

_GLIBCXX_RESOLVE_LIB_DEFECTS 127. `auto_ptr<>` conversion issues These resolutions have all been incorporated.

Definition at line 87 of file `auto_ptr.h`.

5.601.2 Member Typedef Documentation

5.601.2.1 `template<typename _Tp> typedef _Tp std::auto_ptr<_Tp>::element_type`

The pointed-to type.

Definition at line 94 of file `auto_ptr.h`.

5.601.3 Constructor & Destructor Documentation

5.601.3.1 `template<typename _Tp> std::auto_ptr<_Tp>::auto_ptr (element_type *__p = 0) throw () [inline], [explicit]`

An `auto_ptr` is usually constructed from a raw pointer.

Parameters

\leftrightarrow __p	A pointer (defaults to NULL).
--------------------------	-------------------------------

This object now *owns* the object pointed to by __p.

Definition at line 103 of file auto_ptr.h.

5.601.3.2 `template<typename _Tp> std::auto_ptr<_Tp>::auto_ptr(auto_ptr<_Tp> &__a) throw() [inline]`

An auto_ptr can be constructed from another auto_ptr.

Parameters

\leftrightarrow __a	Another auto_ptr of the same type.
--------------------------	------------------------------------

This object now *owns* the object previously owned by __a, which has given up ownership.

Definition at line 112 of file auto_ptr.h.

5.601.3.3 `template<typename _Tp> template<typename _Tp1> std::auto_ptr<_Tp>::auto_ptr(auto_ptr<_Tp1> &__a) throw() [inline]`

An auto_ptr can be constructed from another auto_ptr.

Parameters

\leftrightarrow __a	Another auto_ptr of a different but related type.
--------------------------	---

A pointer-to-Tp1 must be convertible to a pointer-to-Tp/element_type.

This object now *owns* the object previously owned by __a, which has given up ownership.

Definition at line 125 of file auto_ptr.h.

5.601.3.4 `template<typename _Tp> std::auto_ptr<_Tp>::~~auto_ptr() [inline]`

When the auto_ptr goes out of scope, the object it owns is deleted. If it no longer owns anything (i.e., `get()` is NULL), then this has no effect.

The C++ standard says there is supposed to be an empty throw specification here, but omitting it is standard conforming. Its presence can be detected only if `_Tp::~~Tp()` throws, but this is prohibited. [17.4.3.6]/2

Definition at line 170 of file auto_ptr.h.

5.601.3.5 `template<typename _Tp> std::auto_ptr<_Tp>::auto_ptr (auto_ptr_ref< element_type > __ref) throw)`
[inline]

Automatic conversions.

These operations convert an auto_ptr into and from an auto_ptr_ref automatically as needed. This allows constructs such as

```
auto_ptr<Derived> func_returning_auto_ptr(....);  
...  
auto_ptr<Base> ptr = func_returning_auto_ptr(....);
```

Definition at line 260 of file auto_ptr.h.

References std::shared_ptr<_Tp>::shared_ptr(), and std::unique_ptr<_Tp, _Dp>::unique_ptr().

5.601.4 Member Function Documentation

5.601.4.1 `template<typename _Tp> element_type* std::auto_ptr<_Tp>::get (void) const throw)` [inline]

Bypassing the smart pointer.

Returns

The raw pointer being managed.

You can get a copy of the pointer that this object owns, for situations such as passing to a function which only accepts a raw pointer.

Note

This auto_ptr still owns the memory.

Definition at line 211 of file auto_ptr.h.

5.601.4.2 `template<typename _Tp> element_type& std::auto_ptr<_Tp>::operator* () const throw)` [inline]

Smart pointer dereferencing.

If this auto_ptr no longer owns anything, then this operation will crash. (For a smart pointer, *no longer owns anything* is the same as being a null pointer, and you know what happens when you dereference one of those...)

Definition at line 181 of file auto_ptr.h.

5.601.4.3 `template<typename _Tp> element_type* std::auto_ptr<_Tp>::operator-> () const throw)` [inline]

Smart pointer dereferencing.

This returns the pointer itself, which the language then will automatically cause to be dereferenced.

Definition at line 194 of file auto_ptr.h.

5.601.4.4 `template<typename _Tp> auto_ptr& std::auto_ptr<_Tp>::operator= (auto_ptr<_Tp> & __a) throw)`
[inline]

auto_ptr assignment operator.

Parameters

<code>__a</code>	Another <code>auto_ptr</code> of the same type.
------------------	---

This object now *owns* the object previously owned by `__a`, which has given up ownership. The object that this one *used* to own and track has been deleted.

Definition at line 136 of file `auto_ptr.h`.

```
5.601.4.5  template<typename _Tp> template<typename _Tp1 > auto_ptr& std::auto_ptr<_Tp>::operator=( auto_ptr<
        _Tp1 > &__a ) throw )  [inline]
```

`auto_ptr` assignment operator.

Parameters

<code>__a</code>	Another <code>auto_ptr</code> of a different but related type.
------------------	--

A pointer-to-`Tp1` must be convertible to a pointer-to-`Tp/element_type`.

This object now *owns* the object previously owned by `__a`, which has given up ownership. The object that this one *used* to own and track has been deleted.

Definition at line 154 of file `auto_ptr.h`.

```
5.601.4.6  template<typename _Tp> element_type* std::auto_ptr<_Tp>::release ( ) throw )  [inline]
```

Bypassing the smart pointer.

Returns

The raw pointer being managed.

You can get a copy of the pointer that this object owns, for situations such as passing to a function which only accepts a raw pointer.

Note

This `auto_ptr` no longer owns the memory. When this object goes out of scope, nothing will happen.

Definition at line 225 of file `auto_ptr.h`.

```
5.601.4.7  template<typename _Tp> void std::auto_ptr<_Tp>::reset ( element_type * __p = 0 ) throw )  [inline]
```

Forcibly deletes the managed object.

Parameters

<code>__p</code>	A pointer (defaults to NULL).
------------------	-------------------------------

This object now *owns* the object pointed to by `__p`. The previous object has been deleted.

Definition at line 240 of file `auto_ptr.h`.

The documentation for this class was generated from the following file:

- [auto_ptr.h](#)

5.602 std::auto_ptr_ref< _Tp1 > Struct Template Reference

Public Member Functions

- `auto_ptr_ref` (`_Tp1 *__p`)

Public Attributes

- `_Tp1 * _M_ptr`

5.602.1 Detailed Description

```
template<typename _Tp1>
struct std::auto_ptr_ref< _Tp1 >
```

A wrapper class to provide `auto_ptr` with reference semantics. For example, an `auto_ptr` can be assigned (or constructed from) the result of a function which returns an `auto_ptr` by value.

All the `auto_ptr_ref` stuff should happen behind the scenes.

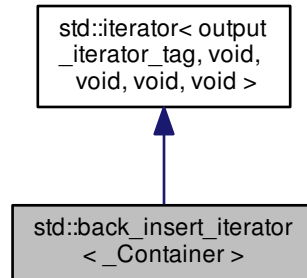
Definition at line 48 of file `auto_ptr.h`.

The documentation for this struct was generated from the following file:

- [auto_ptr.h](#)

5.603 `std::back_insert_iterator<_Container>` Class Template Reference

Inheritance diagram for `std::back_insert_iterator<_Container>`:



Public Types

- typedef `_Container` `container_type`
- typedef void `difference_type`
- typedef `output_iterator_tag` `iterator_category`
- typedef void `pointer`
- typedef void `reference`
- typedef void `value_type`

Public Member Functions

- `back_insert_iterator` (`_Container &__x`)
- `back_insert_iterator` & `operator*` ()
- `back_insert_iterator` & `operator++` ()
- `back_insert_iterator` `operator++` (int)
- `back_insert_iterator` & `operator=` (const typename `_Container::value_type` &__value)
- `back_insert_iterator` & **`operator=`** (typename `_Container::value_type` &&__value)

Protected Attributes

- `_Container *` **`container`**

5.603.1 Detailed Description

```
template<typename _Container>
class std::back_insert_iterator<_Container>
```

Turns assignment into insertion.

These are output iterators, constructed from a container-of-T. Assigning a T to the iterator appends it to the container using push_back.

Tip: Using the back_inserter function to create these iterators can save typing.

Definition at line 449 of file bits/stl_iterator.h.

5.603.2 Member Typedef Documentation

5.603.2.1 `template<typename _Container> typedef _Container std::back_insert_iterator<_Container>::container_type`

A nested typedef for the type of whatever container you used.

Definition at line 457 of file bits/stl_iterator.h.

5.603.2.2 `typedef void std::iterator< output_iterator_tag, void, void, void, void>::difference_type` [inherited]

Distance between iterators is represented as this type.

Definition at line 125 of file stl_iterator_base_types.h.

5.603.2.3 `typedef output_iterator_tag std::iterator< output_iterator_tag, void, void, void, void>::iterator_category` [inherited]

One of the [tag types](#).

Definition at line 121 of file stl_iterator_base_types.h.

5.603.2.4 `typedef void std::iterator< output_iterator_tag, void, void, void, void>::pointer` [inherited]

This type represents a pointer-to-value_type.

Definition at line 127 of file stl_iterator_base_types.h.

5.603.2.5 `typedef void std::iterator< output_iterator_tag, void, void, void, void>::reference` [inherited]

This type represents a reference-to-value_type.

Definition at line 129 of file stl_iterator_base_types.h.

5.603.2.6 `typedef void std::iterator< output_iterator_tag, void, void, void, void >::value_type` [inherited]

The type "pointed to" by the iterator.

Definition at line 123 of file `stl_iterator_base_types.h`.

5.603.3 Constructor & Destructor Documentation

5.603.3.1 `template<typename _Container> std::back_insert_iterator<_Container>::back_insert_iterator(_Container &__x)` [inline], [explicit]

The only way to create this iterator is with a container.

Definition at line 461 of file `bits/stl_iterator.h`.

5.603.4 Member Function Documentation

5.603.4.1 `template<typename _Container> back_insert_iterator& std::back_insert_iterator<_Container>::operator*()` [inline]

Simply returns `*this`.

Definition at line 500 of file `bits/stl_iterator.h`.

5.603.4.2 `template<typename _Container> back_insert_iterator& std::back_insert_iterator<_Container>::operator++()` [inline]

Simply returns `*this`. (This iterator does not *move*.)

Definition at line 505 of file `bits/stl_iterator.h`.

5.603.4.3 `template<typename _Container> back_insert_iterator std::back_insert_iterator<_Container>::operator++(int)` [inline]

Simply returns `*this`. (This iterator does not *move*.)

Definition at line 510 of file `bits/stl_iterator.h`.

5.603.4.4 `template<typename _Container> back_insert_iterator& std::back_insert_iterator<_Container>::operator=(const typename _Container::value_type &__value)` [inline]

Parameters

<code>__value</code>	An instance of whatever type <code>container_type::const_reference</code> is; presumably a reference-to-const T for <code>container<T></code> .
----------------------	---

Returns

This iterator, for chained operations.

This kind of iterator doesn't really have a *position* in the container (you can think of the position as being permanently at the end, if you like). Assigning a value to the iterator will always append the value to the end of the container.

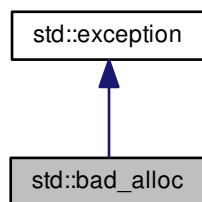
Definition at line 484 of file `bits/stl_iterator.h`.

The documentation for this class was generated from the following file:

- [bits/stl_iterator.h](#)

5.604 `std::bad_alloc` Class Reference

Inheritance diagram for `std::bad_alloc`:



Public Member Functions

- virtual const char * [what](#) () const throw ()

5.604.1 Detailed Description

Exception possibly thrown by `new`.

`bad_alloc` (or classes derived from it) is used to report allocation errors from the throwing forms of `new`.

Definition at line 54 of file `new`.

5.604.2 Member Function Documentation

5.604.2.1 `virtual const char* std::bad_alloc::what () const throw ()` [virtual]

Returns a C-style character string describing the general cause of the current error.

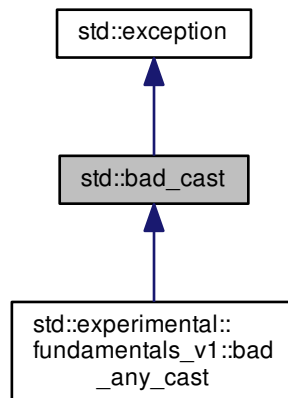
Reimplemented from [std::exception](#).

The documentation for this class was generated from the following file:

- [new](#)

5.605 std::bad_cast Class Reference

Inheritance diagram for `std::bad_cast`:



Public Member Functions

- `virtual const char * what () const noexcept`

5.605.1 Detailed Description

Thrown during incorrect typecasting.

If you attempt an invalid `dynamic_cast` expression, an instance of this class (or something derived from this class) is thrown.

Definition at line 187 of file `typeinfo`.

5.605.2 Member Function Documentation

5.605.2.1 `virtual const char* std::bad_cast::what () const` `[virtual], [noexcept]`

Returns a C-style character string describing the general cause of the current error.

Reimplemented from [std::exception](#).

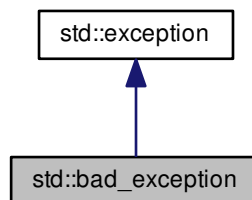
Reimplemented in [std::experimental::fundamentals_v1::bad_any_cast](#).

The documentation for this class was generated from the following file:

- [typeinfo](#)

5.606 `std::bad_exception` Class Reference

Inheritance diagram for `std::bad_exception`:



Public Member Functions

- `virtual const char * what () const` `_GLIBCXX_TXN_SAFE_DYN` `noexcept`

5.606.1 Detailed Description

If an exception is thrown which is not listed in a function's exception specification, one of these may be thrown.

Definition at line 74 of file `exception`.

5.606.2 Member Function Documentation

5.606.2.1 `virtual const char* std::bad_exception::what () const` `[virtual],[noexcept]`

Returns a C-style character string describing the general cause of the current error.

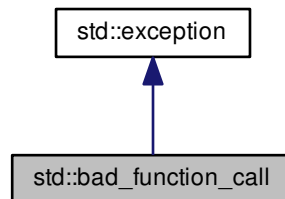
Reimplemented from [std::exception](#).

The documentation for this class was generated from the following file:

- [exception](#)

5.607 `std::bad_function_call` Class Reference

Inheritance diagram for `std::bad_function_call`:



Public Member Functions

- `const char * what () const` `noexcept`

5.607.1 Detailed Description

Exception class thrown when class template function's `operator()` is called with an empty target.

Definition at line 1427 of file `functional`.

5.607.2 Member Function Documentation

5.607.2.1 `const char* std::bad_function_call::what () const` `[virtual],[noexcept]`

Returns a C-style character string describing the general cause of the current error.

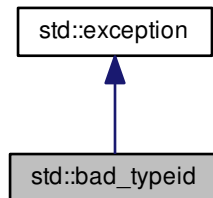
Reimplemented from [std::exception](#).

The documentation for this class was generated from the following file:

- [functional](#)

5.608 `std::bad_typeid` Class Reference

Inheritance diagram for `std::bad_typeid`:



Public Member Functions

- virtual const char * [what](#) () const noexcept

5.608.1 Detailed Description

Thrown when a NULL pointer in a `typeid` expression is used.

Definition at line 204 of file `typeinfo`.

5.608.2 Member Function Documentation

5.608.2.1 virtual const char* `std::bad_typeid::what` () const [virtual], [noexcept]

Returns a C-style character string describing the general cause of the current error.

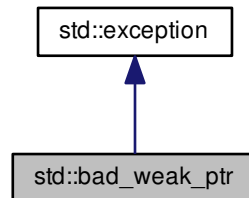
Reimplemented from [std::exception](#).

The documentation for this class was generated from the following file:

- [typeinfo](#)

5.609 `std::bad_weak_ptr` Class Reference

Inheritance diagram for `std::bad_weak_ptr`:



Public Member Functions

- virtual char const * [what](#) () const noexcept

5.609.1 Detailed Description

Exception possibly thrown by `shared_ptr`.

Definition at line 68 of file `shared_ptr_base.h`.

5.609.2 Member Function Documentation

5.609.2.1 virtual char const* `std::bad_weak_ptr::what` () const [virtual], [noexcept]

Returns a C-style character string describing the general cause of the current error.

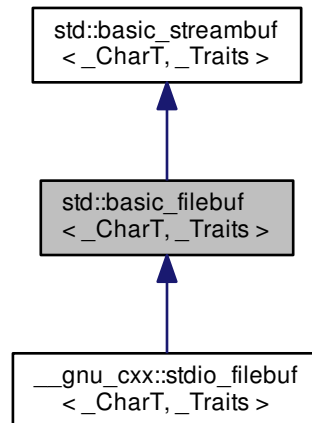
Reimplemented from [std::exception](#).

The documentation for this class was generated from the following file:

- [shared_ptr_base.h](#)

5.610 std::basic_filebuf<_CharT,_Traits> Class Template Reference

Inheritance diagram for std::basic_filebuf<_CharT,_Traits>:



Public Types

- typedef [codecvt](#)< char_type, char, __state_type > **__codecvt_type**
- typedef __basic_file< char > **__file_type**
- typedef [basic_filebuf](#)< char_type, traits_type > **__filebuf_type**
- typedef traits_type::state_type **__state_type**
- typedef [basic_streambuf](#)< char_type, traits_type > **__streambuf_type**
- typedef _CharT **char_type**
- typedef traits_type::int_type **int_type**
- typedef traits_type::off_type **off_type**
- typedef traits_type::pos_type **pos_type**
- typedef _Traits **traits_type**

Public Member Functions

- [basic_filebuf](#) ()
- **basic_filebuf** (const [basic_filebuf](#) &)=delete
- **basic_filebuf** ([basic_filebuf](#) &&)
- virtual [~basic_filebuf](#) ()
- [__filebuf_type](#) * [close](#) ()
- [locale](#) [getloc](#) () const
- [streamsize](#) [in_avail](#) ()
- bool [is_open](#) () const throw ()
- [__filebuf_type](#) * [open](#) (const char *__s, [ios_base::openmode](#) __mode)

- `__filebuf_type * open` (const `std::string` &__s, `ios_base::openmode` __mode)
- `basic_filebuf` & `operator=` (const `basic_filebuf` &)=delete
- `basic_filebuf` & `operator=` (`basic_filebuf` &&)
- `locale pubimbue` (const `locale` &__loc)
- `int_type sbumpc` ()
- `int_type sgetc` ()
- `streamsize sgetn` (char_type *__s, `streamsize` __n)
- `int_type snextc` ()
- `int_type sputbackc` (char_type __c)
- `int_type sputc` (char_type __c)
- `streamsize sputn` (const char_type *__s, `streamsize` __n)
- `int_type sungetc` ()
- void `swap` (`basic_filebuf` &)
- `basic_streambuf * pubsetbuf` (char_type *__s, `streamsize` __n)
- `pos_type pubseekoff` (off_type __off, `ios_base::seekdir` __way, `ios_base::openmode` __mode=`ios_base::in`|`ios_base::out`)
- `pos_type pubseekpos` (pos_type __sp, `ios_base::openmode` __mode=`ios_base::in`|`ios_base::out`)
- `int pubsync` ()

Protected Member Functions

- void `__safe_gbump` (`streamsize` __n)
- void `__safe_pbump` (`streamsize` __n)
- void `_M_allocate_internal_buffer` ()
- bool `_M_convert_to_external` (char_type *, `streamsize`)
- void `_M_create_pback` ()
- void `_M_destroy_internal_buffer` () throw ()
- void `_M_destroy_pback` () throw ()
- int `_M_get_ext_pos` (__state_type &__state)
- `pos_type _M_seek` (off_type __off, `ios_base::seekdir` __way, __state_type __state)
- void `_M_set_buffer` (`streamsize` __off)
- bool `_M_terminate_output` ()
- void `gbump` (int __n)
- virtual void `imbue` (const `locale` &__loc)
- virtual `int_type overflow` (int_type __c=_Traits::eof())
- virtual `int_type pbackfail` (int_type __c=_Traits::eof())
- void `pbump` (int __n)
- virtual `pos_type seekoff` (off_type __off, `ios_base::seekdir` __way, `ios_base::openmode` __mode=`ios_base::in`|`ios_base::out`)
- virtual `pos_type seekpos` (pos_type __pos, `ios_base::openmode` __mode=`ios_base::in`|`ios_base::out`)
- virtual `__streambuf_type * setbuf` (char_type *__s, `streamsize` __n)
- void `setg` (char_type *__gbeg, char_type *__gnext, char_type *__gend)
- void `setp` (char_type *__pbeg, char_type *__pend)
- virtual `streamsize showmanyc` ()
- void `swap` (`basic_streambuf` &__sb)
- virtual `int sync` ()
- virtual `int_type uflow` ()
- virtual `int_type underflow` ()

- virtual [streamsize xsgetn](#) (char_type *__s, [streamsize __n](#))
- virtual [streamsize xspn](#) (const char_type *__s, [streamsize __n](#))
- char_type * [eback](#) () const
- char_type * [gptr](#) () const
- char_type * [egptr](#) () const
- char_type * [pbase](#) () const
- char_type * [pptr](#) () const
- char_type * [epptr](#) () const

Protected Attributes

- char_type * [_M_buf](#)
- bool [_M_buf_allocated](#)
- [locale _M_buf_locale](#)
- size_t [_M_buf_size](#)
- const [__codecvt_type](#) * [_M_codecvt](#)
- char * [_M_ext_buf](#)
- [streamsize _M_ext_buf_size](#)
- char * [_M_ext_end](#)
- const char * [_M_ext_next](#)
- [__file_type _M_file](#)
- char_type * [_M_in_beg](#)
- char_type * [_M_in_cur](#)
- char_type * [_M_in_end](#)
- [__c_lock _M_lock](#)
- [ios_base::openmode _M_mode](#)
- char_type * [_M_out_beg](#)
- char_type * [_M_out_cur](#)
- char_type * [_M_out_end](#)
- bool [_M_reading](#)
- [__state_type _M_state_beg](#)
- [__state_type _M_state_cur](#)
- [__state_type _M_state_last](#)
- bool [_M_writing](#)
- char_type [_M_pback](#)
- char_type * [_M_pback_cur_save](#)
- char_type * [_M_pback_end_save](#)
- bool [_M_pback_init](#)

Friends

- class [ios_base](#)

5.610.1 Detailed Description

```
template<typename _CharT, typename _Traits>
class std::basic_filebuf<_CharT, _Traits>
```

The actual work of input and output (for files).

Template Parameters

<code>_CharT</code>	Type of character stream.
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .

This class associates both its input and output sequence with an external disk file, and maintains a joint file position for both sequences. Many of its semantics are described in terms of similar behavior in the Standard C Library's `FILE` streams.

Requirements on `traits_type`, specific to this class:

- `traits_type::pos_type` must be `fpos<traits_type::state_type>`
- `traits_type::off_type` must be `streamoff`
- `traits_type::state_type` must be Assignable and DefaultConstructible,
- `traits_type::state_type()` must be the initial state for `codecvt`.

Definition at line 72 of file `fstream`.

5.610.2 Constructor & Destructor Documentation

5.610.2.1 `template<typename _CharT, typename _Traits> std::basic_filebuf<_CharT, _Traits>::basic_filebuf ()`

Does not open any files.

The default constructor initializes the parent class using its own default ctor.

Definition at line 80 of file `fstream.tcc`.

References `std::basic_filebuf<_CharT, _Traits>::_M_buf`, `std::basic_streambuf<_CharT, _Traits>::_M_buf_locale`, `std::basic_filebuf<_CharT, _Traits>::_M_buf_size`, `std::basic_filebuf<_CharT, _Traits>::_M_ext_buf`, `std::basic_filebuf<_CharT, _Traits>::_M_ext_buf_size`, `std::basic_filebuf<_CharT, _Traits>::_M_ext_next`, `std::basic_filebuf<_CharT, _Traits>::_M_mode`, `std::basic_filebuf<_CharT, _Traits>::_M_pback`, `std::basic_filebuf<_CharT, _Traits>::_M_pback_cur_save`, `std::basic_filebuf<_CharT, _Traits>::_M_pback_end_save`, `std::basic_filebuf<_CharT, _Traits>::_M_pback_init`, `std::basic_filebuf<_CharT, _Traits>::_M_reading`, `std::basic_filebuf<_CharT, _Traits>::close()`, and `std::basic_filebuf<_CharT, _Traits>::open()`.

5.610.2.2 `template<typename _CharT, typename _Traits> virtual std::basic_filebuf<_CharT, _Traits>::~basic_filebuf ()`
`[inline], [virtual]`

The destructor closes the file first.

Definition at line 238 of file `fstream`.

5.610.3 Member Function Documentation

5.610.3.1 `template<typename _CharT, typename _Traits> void std::basic_filebuf< _CharT, _Traits >::_M_create_pback ()`
`[inline], [protected]`

Initializes pback buffers, and moves normal buffers to safety. Assumptions: `_M_in_cur` has already been moved back

Definition at line 191 of file `fstream`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::pbackfail()`.

5.610.3.2 `template<typename _CharT, typename _Traits> void std::basic_filebuf< _CharT, _Traits >::_M_destroy_pback ()`
`throw) [inline], [protected]`

Deactivates pback buffer contents, and restores normal buffer. Assumptions: The pback buffer has only moved forward.

Definition at line 208 of file `fstream`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_filebuf< _CharT, _Traits >::seekoff()`, `std::basic_filebuf< _CharT, _Traits >::seekpos()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

5.610.3.3 `template<typename _CharT, typename _Traits> void std::basic_filebuf< _CharT, _Traits >::_M_set_buffer (`
`streamsize __off) [inline], [protected]`

This function sets the pointers of the internal buffer, both get and put areas. Typically:

`__off == egptr() - eback()` upon underflow/uflow (**read** mode); `__off == 0` upon overflow (**write** mode); `__off == -1` upon open, setbuf, seekoff/pos (**uncommitted** mode).

NB: `eptr() - pbase() == _M_buf_size - 1`, since `_M_buf_size` reflects the actual allocated memory and the last cell is reserved for the overflow char of a full put area.

Definition at line 422 of file `fstream`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::close()`, `std::basic_filebuf< _CharT, _Traits >::imbue()`, `std::basic_filebuf< _CharT, _Traits >::open()`, `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, `std::basic_filebuf< _CharT, _Traits >::seekpos()`, `__gnu_cxx::stdio_filebuf< _CharT, _Traits >::stdio_filebuf()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, `std::basic_filebuf< _CharT, _Traits >::xsgetn()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

5.610.3.4 `template<typename _CharT, typename _Traits> basic_filebuf< _CharT, _Traits >::__filebuf_type *`
`std::basic_filebuf< _CharT, _Traits >::close ()`

Closes the currently associated file.

Returns

`this` on success, NULL on failure

If no file is currently open, this function immediately fails.

If a *put buffer area* exists, `overflow(eof)` is called to flush all the characters. The file is then closed.

If any operations fail, this function also fails.

Definition at line 213 of file `fstream.tcc`.

References `std::basic_filebuf< _CharT, _Traits >::_M_mode`, `std::basic_filebuf< _CharT, _Traits >::_M_pback_init`, `std::basic_filebuf< _CharT, _Traits >::_M_reading`, `std::basic_filebuf< _CharT, _Traits >::_M_set_buffer()`, `std::basic_filebuf< _CharT, _Traits >::is_open()`, and `std::basic_filebuf< _CharT, _Traits >::showmanyc()`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::basic_filebuf()`, and `std::basic_filebuf< _CharT, _Traits >::open()`.

5.610.3.5 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::eback ()`
`const [inline], [protected], [inherited]`

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Definition at line 482 of file `streambuf`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::imbue()`, `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, `std::basic_filebuf< _CharT, _Traits >::seekpos()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

5.610.3.6 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::egptr ()`
`const [inline], [protected], [inherited]`

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Definition at line 488 of file `streambuf`.

Referenced by `std::ostreambuf_iterator< _CharT, _Traits >::failed()`, `std::basic_filebuf< _CharT, _Traits >::seekpos()`, `std::basic_filebuf< _CharT, _Traits >::showmanyc()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

5.610.3.7 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::eptr ()`
`const [inline], [protected], [inherited]`

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

Definition at line 535 of file `streambuf`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::xspn()`.

5.610.3.8 `template<typename _CharT, typename _Traits> void std::basic_streambuf< _CharT, _Traits >::gbump (int __n)`
`[inline], [protected], [inherited]`

Moving the read position.

Parameters

<code>_↔</code>	The delta by which to move.
<code>_n</code>	

This just advances the read position without returning any data.

Definition at line 498 of file streambuf.

Referenced by `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

5.610.3.9 `template<typename _CharT, typename _Traits> locale std::basic_streambuf< _CharT, _Traits >::getloc () const`
`[inline], [inherited]`

Locale access.

Returns

The current locale in effect.

If `pubimbue(loc)` has been called, then the most recent `loc` is returned. Otherwise the global locale in effect at the time of construction is returned.

Definition at line 226 of file streambuf.

5.610.3.10 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::gptr ()`
`const [inline], [protected], [inherited]`

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Definition at line 485 of file streambuf.

Referenced by `std::ostreambuf_iterator< _CharT, _Traits >::failed()`, `std::basic_filebuf< _CharT, _Traits >::imbue()`, `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, `std::basic_filebuf< _CharT, _Traits >::seekpos()`, `std::basic_filebuf< _CharT, _Traits >::showmanyc()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

5.610.3.11 `template<typename _CharT, typename _Traits> void std::basic_filebuf< _CharT, _Traits >::imbue (const locale`
`& __loc) [protected], [virtual]`

Changes translations.

Parameters

<code>__loc</code>	A new locale.
--------------------	---------------

Translations done during I/O which depend on the current locale are changed by this call. The standard adds, *Between invocations of this function a class derived from `streambuf` can safely cache results of calls to locale functions and to members of facets so obtained.*

Note

Base class version does nothing.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

Definition at line 997 of file `fstream.tcc`.

References `std::basic_filebuf<_CharT, _Traits>::_M_ext_buf`, `std::basic_filebuf<_CharT, _Traits>::_M_ext_next`, `std::basic_filebuf<_CharT, _Traits>::_M_mode`, `std::basic_filebuf<_CharT, _Traits>::_M_reading`, `std::basic_filebuf<_CharT, _Traits>::_M_set_buffer()`, `std::ios_base::cur`, `std::basic_streambuf<_CharT, _Traits>::eback()`, `std::basic_streambuf<_CharT, _Traits>::gptr()`, `std::basic_filebuf<_CharT, _Traits>::is_open()`, and `std::basic_filebuf<_CharT, _Traits>::seekoff()`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::sync()`.

```
5.610.3.12 template<typename _CharT, typename _Traits> streamsize std::basic_streambuf<_CharT, _Traits>::in_avail (
    ) [inline], [inherited]
```

Looking ahead into the stream.

Returns

The number of characters available.

If a read position is available, returns the number of characters available for reading before the buffer must be refilled. Otherwise returns the derived `showmanyc()`.

Definition at line 284 of file `streambuf`.

```
5.610.3.13 template<typename _CharT, typename _Traits> bool std::basic_filebuf<_CharT, _Traits>::is_open ( ) const throw
    ) [inline]
```

Returns true if the external file is open.

Definition at line 252 of file `fstream`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::close()`, `std::basic_filebuf<_CharT, _Traits>::imbue()`, `std::basic_filebuf<_CharT, _Traits>::open()`, `std::basic_filebuf<_CharT, _Traits>::seekoff()`, `std::basic_filebuf<_CharT, _Traits>::seekpos()`, `std::basic_filebuf<_CharT, _Traits>::setbuf()`, `std::basic_filebuf<_CharT, _Traits>::showmanyc()`, and `__gnu_cxx::stdio_filebuf<_CharT, _Traits>::stdio_filebuf()`.

```
5.610.3.14 template<typename _CharT, typename _Traits> basic_filebuf<_CharT, _Traits>::__filebuf_type *
    std::basic_filebuf<_CharT, _Traits>::open ( const char * __s, ios_base::openmode __mode )
```

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Returns

`this` on success, NULL on failure

If a file is already open, this function immediately fails. Otherwise it tries to open the file named `__s` using the flags given in `__mode`.

Table 92, adapted here, gives the relation between openmode combinations and the equivalent `fopen()` flags. (NB: lines `app`, `in|out|app`, `in|app`, `binary|app`, `binary|in|out|app`, and `binary|in|app` per DR 596)

ios_base Flag combination					stdio equivalent
binary	in	out	trunc	app	
		+			w
		+		+	a
				+	a
		+	+		w
	+				r
	+	+			r+
	+	+	+		w+
	+	+		+	a+
	+			+	a+
+		+			wb
+		+		+	ab
+				+	ab
+		+	+		wb
+	+				rb
+	+	+			r+b
+	+	+	+		w+b
+	+	+		+	a+b
+	+			+	a+b

Definition at line 179 of file `fstream.tcc`.

References `std::basic_filebuf< _CharT, _Traits >::_M_mode`, `std::basic_filebuf< _CharT, _Traits >::_M_reading`, `std::basic_filebuf< _CharT, _Traits >::_M_set_buffer()`, `std::ios_base::ate`, `std::basic_filebuf< _CharT, _Traits >::close()`, `std::ios_base::end`, `std::basic_filebuf< _CharT, _Traits >::is_open()`, and `std::basic_filebuf< _CharT, _Traits >::seekoff()`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::basic_filebuf()`.

5.610.3.15 `template<typename _CharT, typename _Traits> __filebuf_type* std::basic_filebuf< _CharT, _Traits >::open (const std::string & __s, ios_base::openmode __mode) [inline]`

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Returns

`this` on success, `NULL` on failure

Definition at line 307 of file `fstream`.

5.610.3.16 `template<typename _CharT, typename _Traits> basic_filebuf<_CharT, _Traits>::int_type std::basic_filebuf<_CharT, _Traits>::overflow(int_type __c = _Traits::eof()) [protected], [virtual]`

Consumes data from the buffer; writes to the controlled sequence.

Parameters

<code>__c</code>	An additional character to consume.
------------------	-------------------------------------

Returns

`eof()` to indicate failure, something else (usually `__c`, or `not_eof()`)

Informally, this function is called when the output buffer is full (or does not exist, as buffering need not actually be done). If a buffer exists, it is *consumed*, with *some effect* on the controlled sequence. (Typically, the buffer is written out to the sequence verbatim.) In either case, the character `c` is also written out, if `__c` is not `eof()`.

For a formal definition of this function, see a good text such as Langer & Kreft, or [27.5.2.4.5]/3-7.

A functioning output streambuf can be created by overriding only this function (no buffer area will be used).

Note

Base class version does nothing, returns `eof()`.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

Definition at line 507 of file `fstream.tcc`.

References `std::basic_filebuf<_CharT, _Traits>::M_buf_size`, `std::basic_filebuf<_CharT, _Traits>::M_destroy`, `std::basic_filebuf<_CharT, _Traits>::M_mode`, `std::basic_filebuf<_CharT, _Traits>::M_reading`, `std::basic_filebuf<_CharT, _Traits>::M_set_buffer()`, `std::ios_base::app`, `std::ios_base::cur`, `std::ios_base::out`, `std::basic_streambuf<_CharT, _Traits>::pbase()`, `std::basic_streambuf<_CharT, _Traits>::pbump()`, `std::basic_streambuf<_CharT, _Traits>::pptr()`, `std::basic_filebuf<_CharT, _Traits>::xsgetn()`, and `std::basic_filebuf<_CharT, _Traits>::xspn()`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::pbackfail()`, `std::basic_filebuf<_CharT, _Traits>::seekpos()`, `std::basic_filebuf<_CharT, _Traits>::sync()`, `std::basic_filebuf<_CharT, _Traits>::underflow()`, and `std::basic_filebuf<_CharT, _Traits>::xsgetn()`.

5.610.3.17 `template<typename _CharT, typename _Traits> basic_filebuf<_CharT,_Traits>::int_type std::basic_filebuf<_CharT,_Traits>::pbackfail (int_type __c = _Traits::eof()) [protected],[virtual]`

Tries to back up the input sequence.

Parameters

<code>_↔</code>	The character to be inserted back into the sequence.
<code>_c</code>	

Returns

`eof()` on failure, *some other value* on success

Postcondition

The constraints of `gptr()`, `eback()`, and `pptr()` are the same as for `underflow()`.

Note

Base class version does nothing, returns `eof()`.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

Definition at line 448 of file `fstream.tcc`.

References `std::basic_filebuf<_CharT, _Traits>::_M_create_pback()`, `std::basic_filebuf<_CharT, _Traits>::_M_↔mode`, `std::basic_filebuf<_CharT, _Traits>::_M_pback_init`, `std::basic_filebuf<_CharT, _Traits>::_M_reading`, `std↔::basic_filebuf<_CharT, _Traits>::_M_set_buffer()`, `std::ios_base::cur`, `std::basic_streambuf<_CharT, _Traits>↔::eback()`, `std::basic_streambuf<_CharT, _Traits>::gbump()`, `std::basic_streambuf<_CharT, _Traits>::gptr()`, `std↔::ios_base::in`, `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_filebuf<_CharT, _Traits>::seekoff()`, and `std::basic_filebuf<_CharT, _Traits>::underflow()`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::underflow()`.

5.610.3.18 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::pbase ()`
`const [inline], [protected], [inherited]`

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

Definition at line 529 of file `streambuf`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_filebuf<_CharT, _Traits>::seekoff()`, `std↔::basic_filebuf<_CharT, _Traits>::seekpos()`, `std::basic_filebuf<_CharT, _Traits>::sync()`, and `std::basic_filebuf<_CharT, _Traits>::xsputn()`.

5.610.3.19 `template<typename _CharT, typename _Traits> void std::basic_streambuf<_CharT, _Traits>::pbump (int __n)`
`[inline], [protected], [inherited]`

Moving the write position.

Parameters

<code>_↔</code>	The delta by which to move.
<code>_n</code>	

This just advances the write position without returning any data.

Definition at line 545 of file streambuf.

Referenced by std::basic_filebuf< _CharT, _Traits >::overflow().

5.610.3.20 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::pptr ()`
`const [inline], [protected], [inherited]`

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- pbase() returns the beginning pointer for the output sequence
- pptr() returns the next pointer for the output sequence
- epptr() returns the end pointer for the output sequence

Definition at line 532 of file streambuf.

Referenced by std::basic_filebuf< _CharT, _Traits >::overflow(), std::basic_filebuf< _CharT, _Traits >::seekoff(), std::basic_filebuf< _CharT, _Traits >::seekpos(), std::basic_filebuf< _CharT, _Traits >::sync(), and std::basic_filebuf< _CharT, _Traits >::xsputn().

5.610.3.21 `template<typename _CharT, typename _Traits> locale std::basic_streambuf< _CharT, _Traits >::pubimbue (`
`const locale &__loc) [inline], [inherited]`

Entry point for imbue().

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls the derived imbue(__loc).

Definition at line 209 of file streambuf.


```
5.610.3.22 template<typename _CharT, typename _Traits> pos_type std::basic_streambuf< _CharT, _Traits >::pubseekoff (
    off_type __off, ios_base::seekdir __way, ios_base::openmode __mode = ios_base::in | ios_base::out )
    [inline], [inherited]
```

Alters the stream position.

Parameters

<code>__off</code>	Offset.
<code>__way</code>	Value for <code>ios_base::seekdir</code> .
<code>__mode</code>	Value for <code>ios_base::openmode</code> .

Calls virtual seekoff function.

Definition at line 251 of file streambuf.

```
5.610.3.23 template<typename _CharT, typename _Traits> pos_type std::basic_streambuf< _CharT, _Traits >::pubseekpos
    ( pos_type __sp, ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline],
    [inherited]
```

Alters the stream position.

Parameters

<code>__sp</code>	Position
<code>__mode</code>	Value for <code>ios_base::openmode</code> .

Calls virtual seekpos function.

Definition at line 263 of file streambuf.

```
5.610.3.24 template<typename _CharT, typename _Traits> basic_streambuf* std::basic_streambuf< _CharT, _Traits
    >::pubsetbuf( char_type *__s, streamsize __n ) [inline], [inherited]
```

Entry points for derived buffer functions.

The public versions of `pubfoo` dispatch to the protected derived `foo` member functions, passing the arguments (if any) and returning the result unchanged.

Definition at line 239 of file streambuf.

```
5.610.3.25 template<typename _CharT, typename _Traits> int std::basic_streambuf< _CharT, _Traits >::pubsync ( )
    [inline], [inherited]
```

Calls virtual sync function.

Definition at line 271 of file streambuf.

Referenced by `std::basic_istream< _CharT, _Traits >::sync()`.

5.610.3.26 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf< _CharT, _Traits >::sbumpc ()`
`[inline], [inherited]`

Getting the next character.

Returns

The next character, or eof.

If the input read position is available, returns that character and increments the read pointer, otherwise calls and returns `uflow()`.

Definition at line 316 of file `streambuf`.

Referenced by `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, and `std::istreambuf_iterator< _CharT, _Traits >::operator++()`.

5.610.3.27 `template<typename _CharT, typename _Traits > basic_filebuf< _CharT, _Traits >::pos_type`
`std::basic_filebuf< _CharT, _Traits >::seekoff (off_type, ios_base::seekdir, ios_base::openmode =`
`ios_base::in | ios_base::out) [protected], [virtual]`

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented from [std::basic_streambuf< _CharT, _Traits >](#).

Definition at line 800 of file `fstream.tcc`.

References `std::basic_filebuf< _CharT, _Traits >::M_destroy_pback()`, `std::basic_filebuf< _CharT, _Traits >::M_↔_reading`, `std::ios_base::cur`, `std::basic_filebuf< _CharT, _Traits >::is_open()`, `std::basic_streambuf< _CharT, _Traits >::pbase()`, `std::basic_streambuf< _CharT, _Traits >::pptr()`, and `std::basic_filebuf< _CharT, _Traits >::seekpos()`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::imbue()`, `std::basic_filebuf< _CharT, _Traits >::open()`, `std::↔_basic_filebuf< _CharT, _Traits >::pbackfail()`, and `std::basic_filebuf< _CharT, _Traits >::setbuf()`.

5.610.3.28 `template<typename _CharT, typename _Traits > basic_filebuf< _CharT, _Traits >::pos_type std::basic_filebuf<`
`_CharT, _Traits >::seekpos (pos_type, ios_base::openmode = ios_base::in | ios_base::out)`
`[protected], [virtual]`

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented from [std::basic_streambuf< _CharT, _Traits >](#).

Definition at line 860 of file `fstream.tcc`.

References `std::basic_filebuf< _CharT, _Traits >::M_destroy_pback()`, `std::basic_filebuf< _CharT, _Traits >::M_↔_ext_buf`, `std::basic_filebuf< _CharT, _Traits >::M_ext_next`, `std::basic_filebuf< _CharT, _Traits >::M_reading`, `std::↔_basic_filebuf< _CharT, _Traits >::M_set_buffer()`, `std::ios_base::beg`, `std::basic_streambuf< _CharT, _Traits >↔_eback()`, `std::basic_streambuf< _CharT, _Traits >::egptr()`, `std::basic_streambuf< _CharT, _Traits >::gpptr()`, `std::↔_basic_filebuf< _CharT, _Traits >::is_open()`, `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_streambuf< _CharT, _Traits >::pbase()`, `std::basic_streambuf< _CharT, _Traits >::pptr()`, and `std::basic_filebuf< _CharT, _Traits >::sync()`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::seekoff()`.

```
5.610.3.29 template<typename _CharT, typename _Traits> basic_filebuf<_CharT, _Traits>::__streambuf_type *
std::basic_filebuf<_CharT, _Traits>::setbuf ( char_type * __s, streamsize __n ) [protected],
[virtual]
```

Manipulates the buffer.

Parameters

<code>__s</code>	Pointer to a buffer area.
<code>__n</code>	Size of <code>__s</code> .

Returns

`this`

If no file has been opened, and both `__s` and `__n` are zero, then the stream becomes unbuffered. Otherwise, `__s` is used as a buffer; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html#io.streambuf.buffering> for more.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

Definition at line 771 of file `fstream.tcc`.

References `std::basic_filebuf<_CharT, _Traits>::_M_buf`, `std::basic_filebuf<_CharT, _Traits>::_M_buf_size`, `std::basic_filebuf<_CharT, _Traits>::is_open()`, and `std::basic_filebuf<_CharT, _Traits>::seekoff()`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::xsputn()`.

```
5.610.3.30 template<typename _CharT, typename _Traits> void std::basic_streambuf<_CharT, _Traits>::setg ( char_type
* __gbeg, char_type * __gnext, char_type * __gend ) [inline], [protected], [inherited]
```

Setting the three read area pointers.

Parameters

<code>__gbeg</code>	A pointer.
<code>__gnext</code>	A pointer.
<code>__gend</code>	A pointer.

Postcondition

`__gbeg == eback()`, `__gnext == gptr()`, and `__gend == egptr()`

Definition at line 509 of file `streambuf`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::xsgetn()`.

5.610.3.31 `template<typename _CharT, typename _Traits> void std::basic_streambuf<_CharT, _Traits>::setp (char_type * __pbeg, char_type * __pend)` [inline], [protected], [inherited]

Setting the three write area pointers.

Parameters

<code>__pbeg</code>	A pointer.
<code>__pend</code>	A pointer.

Postcondition

`__pbeg == pbase()`, `__pbeg == pptr()`, and `__pend == epptr()`

Definition at line 555 of file streambuf.

5.610.3.32 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::sgetc ()` [inline], [inherited]

Getting the next character.

Returns

The next character, or eof.

If the input read position is available, returns that character, otherwise calls and returns `underflow()`. Does not move the read position after fetching the character.

Definition at line 338 of file streambuf.

Referenced by `std::istreambuf_iterator<_CharT, _Traits>::equal()`, `std::ostreambuf_iterator<_CharT, _Traits>::failed()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, and `std::basic_streambuf<_CharT, _Traits>::xsputn()`.

5.610.3.33 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf<_CharT, _Traits>::sgetn (char_type * __s, streamsize __n)` [inline], [inherited]

Entry point for `xsgetn`.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	A count.

Returns `xsgetn(__s,__n)`. The effect is to fill `__s[0]` through `__s[__n-1]` with characters from the input sequence, if possible.

Definition at line 357 of file `streambuf`.

5.610.3.34 `template<typename _CharT, typename _Traits> streamsize std::basic_filebuf<_CharT, _Traits>::showmanyc () [protected],[virtual]`

Investigating the data available.

Returns

An estimate of the number of characters available in the input sequence, or -1.

If it returns a positive value, then successive calls to `underflow()` will not return `traits::eof()` until at least that number of characters have been supplied. If `showmanyc()` returns -1, then calls to `underflow()` or `uflow()` will fail. [27.5.2.4.3]/1

Note

Base class version does nothing, returns zero.

The standard adds that *the intention is not only that the calls [to `underflow` or `uflow`] will not return `eof()` but that they will return immediately.*

The standard adds that *the morphemes of `showmanyc` are **es-how-many-see**, not **show-manic**.*

Reimplemented from `std::basic_streambuf<_CharT, _Traits>`.

Definition at line 263 of file `fstream.tcc`.

References `std::basic_filebuf<_CharT, _Traits>::M_mode`, `std::ios_base::binary`, `std::basic_streambuf<_CharT, _Traits>::egptr()`, `std::basic_streambuf<_CharT, _Traits>::gptr()`, `std::ios_base::in`, `std::basic_filebuf<_CharT, _Traits>::is_open()`, and `std::basic_filebuf<_CharT, _Traits>::underflow()`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::close()`.

5.610.3.35 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::snextc () [inline],[inherited]`

Getting the next character.

Returns

The next character, or `eof`.

Calls `sbumpc()`, and if that function returns `traits::eof()`, so does this function. Otherwise, `sgetc()`.

Definition at line 298 of file `streambuf`.

Referenced by `std::ostreambuf_iterator<_CharT, _Traits>::failed()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, and `std::basic_streambuf<_CharT, _Traits>::xsputn()`.

5.610.3.36 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::sputbackc (char_type _c) [inline],[inherited]`

Pushing characters back into the input stream.

Parameters

<code>__c</code>	The character to push back.
------------------	-----------------------------

Returns

The previous character, if possible.

Similar to `sungetc()`, but `__c` is pushed onto the stream instead of *the previous character*. If successful, the next character fetched from the input stream will be `__c`.

Definition at line 372 of file `streambuf`.

Referenced by `std::basic_istream< _CharT, _Traits >::putback()`.

5.610.3.37 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf< _CharT, _Traits >::sputc (char_type __c) [inline], [inherited]`

Entry point for all single-character output functions.

Parameters

<code>__c</code>	A character to output.
------------------	------------------------

Returns

`__c`, if possible.

One of two public output functions.

If a write position is available for the output sequence (i.e., the buffer is not full), stores `__c` in that position, increments the position, and returns `traits::to_int_type(__c)`. If a write position is not available, returns `overflow(__c)`.

Definition at line 424 of file `streambuf`.

Referenced by `std::basic_istream< _CharT, _Traits >::get()`, `std::ostreambuf_iterator< _CharT, _Traits >::operator=()`, and `std::basic_streambuf< _CharT, _Traits >::xsputn()`.

5.610.3.38 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf< _CharT, _Traits >::sputn (const char_type *__s, streamsize __n) [inline], [inherited]`

Entry point for all single-character output functions.

Parameters

<code>__s</code>	A buffer read area.
<code>__n</code>	A count.

One of two public output functions.

Returns `xspn(__s, __n)`. The effect is to write `__s[0]` through `__s[__n-1]` to the output sequence, if possible.

Definition at line 450 of file `streambuf`.

Referenced by `std::ostreambuf_iterator<_CharT, _Traits>::failed()`.

5.610.3.39 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::sungetc ()`
`[inline], [inherited]`

Moving backwards in the input stream.

Returns

The previous character, if possible.

If a putback position is available, this function decrements the input pointer and returns that character. Otherwise, calls and returns `pbackfail()`. The effect is to *unget* the last character *gotten*.

Definition at line 397 of file `streambuf`.

Referenced by `std::basic_istream<_CharT, _Traits>::unget()`.

5.610.3.40 `template<typename _CharT, typename _Traits> int std::basic_filebuf<_CharT, _Traits>::sync ()`
`[protected], [virtual]`

Synchronizes the buffer arrays with the controlled sequences.

Returns

-1 on failure.

Each derived class provides its own appropriate behavior, including the definition of *failure*.

Note

Base class version does nothing, returns zero.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

Definition at line 980 of file `fstream.tcc`.

References `std::basic_filebuf<_CharT, _Traits>::imbue()`, `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_streambuf<_CharT, _Traits>::pbase()`, and `std::basic_streambuf<_CharT, _Traits>::pptr()`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::seekpos()`.

```
5.610.3.41 template<typename _CharT, typename _Traits> virtual int_type std::basic_streambuf< _CharT, _Traits >::uflow (
    ) [inline], [protected], [virtual], [inherited]
```

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function does the same thing as `underflow()`, and in fact is required to call that function. It also returns the new character, like `underflow()` does. However, this function also moves the read position forward by one.

Reimplemented in [__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >](#).

Definition at line 700 of file `streambuf`.

```
5.610.3.42 template<typename _CharT, typename _Traits > basic_filebuf< _CharT, _Traits >::int_type std::basic_filebuf<
    _CharT, _Traits >::underflow ( ) [protected], [virtual]
```

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function is called when the input buffer is exhausted (or does not exist, as buffering need not actually be done). If a buffer exists, it is *refilled*. In either case, the next available character is returned, or `traits::eof()` to indicate a null pending sequence.

For a formal definition of the pending sequence, see a good text such as Langer & Kreft, or [27.5.2.4.3]/7-14.

A functioning input streambuf can be created by overriding only this function (no buffer area will be used). For an example, see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html>

Note

Base class version does nothing, returns `eof()`.

Reimplemented from [std::basic_streambuf< _CharT, _Traits >](#).

Definition at line 289 of file `fstream.tcc`.

References `std::basic_filebuf< _CharT, _Traits >::M_buf_size`, `std::basic_filebuf< _CharT, _Traits >::M_destroy`, `std::basic_filebuf< _CharT, _Traits >::M_ext_buf`, `std::basic_filebuf< _CharT, _Traits >::M_ext_buf_size`, `std::basic_filebuf< _CharT, _Traits >::M_ext_next`, `std::basic_filebuf< _CharT, _Traits >::M_mode`, `std::basic_filebuf< _CharT, _Traits >::M_reading`, `std::basic_filebuf< _CharT, _Traits >::M_set_buffer()`, `std::basic_streambuf< _CharT, _Traits >::eback()`, `std::basic_streambuf< _CharT, _Traits >::egptr()`, `std::basic_streambuf< _CharT, _Traits >::gptr()`, `std::ios_base::in`, `std::min()`, `std::basic_filebuf< _CharT, _Traits >::overflow()`, and `std::basic_filebuf< _CharT, _Traits >::pbackfail()`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, and `std::basic_filebuf< _CharT, _Traits >::showmanyc()`.

```
5.610.3.43 template<typename _CharT, typename _Traits > streamsize std::basic_filebuf< _CharT, _Traits >::xsgetn (
    char_type * __s, streamsize __n ) [protected], [virtual]
```

Multiple character extraction.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to assign.

Returns

The number of characters assigned.

Fills `__s[0]` through `__s[__n-1]` with characters from the input sequence, as if by `sbumpc()`. Stops when either `__n` characters have been copied, or when `traits::eof()` would be copied.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

Definition at line 635 of file `fstream.tcc`.

References `std::basic_filebuf<_CharT, _Traits>::M_buf_size`, `std::basic_filebuf<_CharT, _Traits>::M_destroy`, `std::basic_filebuf<_CharT, _Traits>::M_mode`, `std::basic_filebuf<_CharT, _Traits>::M_pback_init`, `std::basic_filebuf<_CharT, _Traits>::M_reading`, `std::basic_filebuf<_CharT, _Traits>::M_set_buffer()`, `std::basic_streambuf<_CharT, _Traits>::eback()`, `std::basic_streambuf<_CharT, _Traits>::egptr()`, `std::basic_streambuf<_CharT, _Traits>::gbump()`, `std::basic_streambuf<_CharT, _Traits>::gptr()`, `std::ios_base::in`, `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_streambuf<_CharT, _Traits>::setg()`, `std::basic_streambuf<_CharT, _Traits>::xsgetn()`, and `std::basic_filebuf<_CharT, _Traits>::xspn()`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::overflow()`.

5.610.3.44 `template<typename _CharT, typename _Traits> streamsize std::basic_filebuf<_CharT, _Traits>::xspn (const char_type * __s, streamsize __n) [protected], [virtual]`

Multiple character insertion.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to write.

Returns

The number of characters written.

Writes `__s[0]` through `__s[__n-1]` to the output sequence, as if by `sputc()`. Stops when either `n` characters have been copied, or when `sputc()` would return `traits::eof()`.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented from [std::basic_streambuf< _CharT, _Traits >](#).

Definition at line 723 of file fstream.tcc.

References [std::basic_filebuf< _CharT, _Traits >::_M_buf_size](#), [std::basic_filebuf< _CharT, _Traits >::_M_mode](#), [std::basic_filebuf< _CharT, _Traits >::_M_reading](#), [std::basic_filebuf< _CharT, _Traits >::_M_set_buffer\(\)](#), [std::ios_base::app](#), [std::basic_streambuf< _CharT, _Traits >::epptr\(\)](#), [std::min\(\)](#), [std::ios_base::out](#), [std::basic_streambuf< _CharT, _Traits >::pbase\(\)](#), [std::basic_streambuf< _CharT, _Traits >::pptr\(\)](#), [std::basic_filebuf< _CharT, _Traits >::setbuf\(\)](#), and [std::basic_streambuf< _CharT, _Traits >::xsputn\(\)](#).

Referenced by [std::basic_filebuf< _CharT, _Traits >::overflow\(\)](#), and [std::basic_filebuf< _CharT, _Traits >::xsgetn\(\)](#).

5.610.4 Member Data Documentation

5.610.4.1 `template<typename _CharT, typename _Traits> char_type* std::basic_filebuf< _CharT, _Traits >::_M_buf`
[protected]

Pointer to the beginning of internal buffer.

Definition at line 128 of file fstream.

Referenced by [std::basic_filebuf< _CharT, _Traits >::basic_filebuf\(\)](#), and [std::basic_filebuf< _CharT, _Traits >::setbuf\(\)](#).

5.610.4.2 `template<typename _CharT, typename _Traits> locale std::basic_streambuf< _CharT, _Traits >::_M_buf_locale`
[protected], [inherited]

Current locale setting.

Definition at line 192 of file streambuf.

Referenced by [std::basic_filebuf< _CharT, _Traits >::basic_filebuf\(\)](#).

5.610.4.3 `template<typename _CharT, typename _Traits> size_t std::basic_filebuf< _CharT, _Traits >::_M_buf_size`
[protected]

Actual size of internal buffer. This number is equal to the size of the put area + 1 position, reserved for the overflow char of a full area.

Definition at line 135 of file fstream.

Referenced by [std::basic_filebuf< _CharT, _Traits >::basic_filebuf\(\)](#), [std::basic_filebuf< _CharT, _Traits >::overflow\(\)](#), [std::basic_filebuf< _CharT, _Traits >::setbuf\(\)](#), [__gnu_cxx::stdio_filebuf< _CharT, _Traits >::stdio_filebuf\(\)](#), [std::basic_filebuf< _CharT, _Traits >::underflow\(\)](#), [std::basic_filebuf< _CharT, _Traits >::xsgetn\(\)](#), and [std::basic_filebuf< _CharT, _Traits >::xsputn\(\)](#).

5.610.4.4 `template<typename _CharT, typename _Traits> char* std::basic_filebuf< _CharT, _Traits >::_M_ext_buf`
`[protected]`

Buffer for external characters. Used for input when `codecvt::always_noconv() == false`. When valid, this corresponds to `eback()`.

Definition at line 170 of file `fstream`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::basic_filebuf()`, `std::basic_filebuf< _CharT, _Traits >::imbue()`, `std::basic_filebuf< _CharT, _Traits >::seekpos()`, and `std::basic_filebuf< _CharT, _Traits >::underflow()`.

5.610.4.5 `template<typename _CharT, typename _Traits> streamsize std::basic_filebuf< _CharT, _Traits >::_M_ext_buf_size`
`[protected]`

Size of buffer held by `_M_ext_buf`.

Definition at line 175 of file `fstream`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::basic_filebuf()`, and `std::basic_filebuf< _CharT, _Traits >::underflow()`.

5.610.4.6 `template<typename _CharT, typename _Traits> const char* std::basic_filebuf< _CharT, _Traits >::_M_ext_next`
`[protected]`

Pointers into the buffer held by `_M_ext_buf` that delimit a subsequence of bytes that have been read but not yet converted. When valid, `_M_ext_next` corresponds to `egptr()`.

Definition at line 182 of file `fstream`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::basic_filebuf()`, `std::basic_filebuf< _CharT, _Traits >::imbue()`, `std::basic_filebuf< _CharT, _Traits >::seekpos()`, and `std::basic_filebuf< _CharT, _Traits >::underflow()`.

5.610.4.7 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::_M_in_beg`
`[protected], [inherited]`

Start of get area.

Definition at line 184 of file `streambuf`.

5.610.4.8 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::_M_in_cur`
`[protected], [inherited]`

Current read area.

Definition at line 185 of file `streambuf`.

5.610.4.9 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::_M_in_end`
`[protected], [inherited]`

End of get area.

Definition at line 186 of file `streambuf`.

5.610.4.10 `template<typename _CharT, typename _Traits> ios_base::openmode std::basic_filebuf< _CharT, _Traits >::_M_mode [protected]`

Place to stash in || out || in | out settings for current filebuf.

Definition at line 113 of file fstream.

Referenced by `std::basic_filebuf< _CharT, _Traits >::basic_filebuf()`, `std::basic_filebuf< _CharT, _Traits >::close()`, `std::basic_filebuf< _CharT, _Traits >::imbue()`, `std::basic_filebuf< _CharT, _Traits >::open()`, `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, `std::basic_filebuf< _CharT, _Traits >::showmanyc()`, `__gnu_cxx::stdio_filebuf< _CharT, _Traits >::stdio_filebuf()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, `std::basic_filebuf< _CharT, _Traits >::xsgetn()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

5.610.4.11 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_beg [protected], [inherited]`

Start of put area.

Definition at line 187 of file streambuf.

5.610.4.12 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_cur [protected], [inherited]`

Current put area.

Definition at line 188 of file streambuf.

5.610.4.13 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_end [protected], [inherited]`

End of put area.

Definition at line 189 of file streambuf.

5.610.4.14 `template<typename _CharT, typename _Traits> char_type std::basic_filebuf< _CharT, _Traits >::_M_pback [protected]`

Necessary bits for putback buffer management.

Note

pbacks of over one character are not currently supported.

Definition at line 156 of file fstream.

Referenced by `std::basic_filebuf< _CharT, _Traits >::basic_filebuf()`.

5.610.4.15 `template<typename _CharT, typename _Traits> char_type* std::basic_filebuf< _CharT, _Traits >::_M_pback_cur_save [protected]`

Necessary bits for putback buffer management.

Note

pbacks of over one character are not currently supported.

Definition at line 157 of file `fstream`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::basic_filebuf()`.

5.610.4.16 `template<typename _CharT, typename _Traits> char_type* std::basic_filebuf< _CharT, _Traits >::_M_pback_end_save [protected]`

Necessary bits for putback buffer management.

Note

pbacks of over one character are not currently supported.

Definition at line 158 of file `fstream`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::basic_filebuf()`.

5.610.4.17 `template<typename _CharT, typename _Traits> bool std::basic_filebuf< _CharT, _Traits >::_M_pback_init [protected]`

Necessary bits for putback buffer management.

Note

pbacks of over one character are not currently supported.

Definition at line 159 of file `fstream`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::basic_filebuf()`, `std::basic_filebuf< _CharT, _Traits >::close()`, `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

5.610.4.18 `template<typename _CharT, typename _Traits> bool std::basic_filebuf< _CharT, _Traits >::_M_reading [protected]`

`_M_reading == false` && `_M_writing == false` for **uncommitted** mode; `_M_reading == true` for **read** mode; `_M_writing == true` for **write** mode;

NB: `_M_reading == true` && `_M_writing == true` is unused.

Definition at line 147 of file `fstream`.

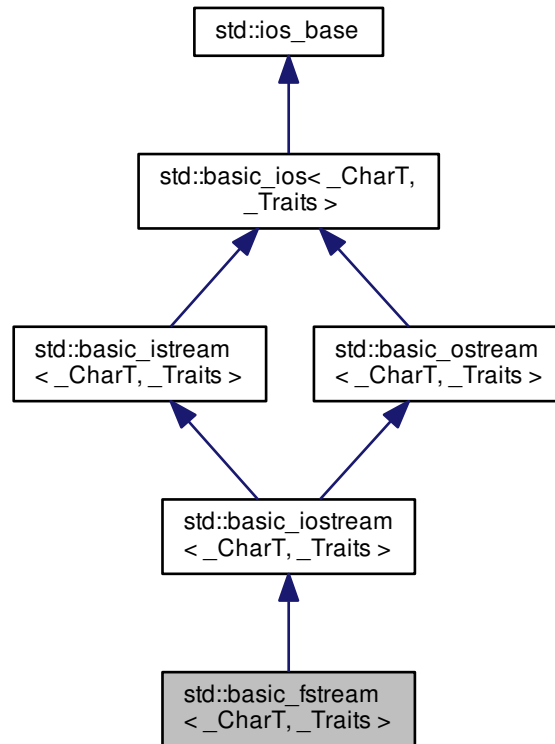
Referenced by `std::basic_filebuf< _CharT, _Traits >::basic_filebuf()`, `std::basic_filebuf< _CharT, _Traits >::close()`, `std::basic_filebuf< _CharT, _Traits >::imbue()`, `std::basic_filebuf< _CharT, _Traits >::open()`, `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, `std::basic_filebuf< _CharT, _Traits >::seekoff()`, `std::basic_filebuf< _CharT, _Traits >::seekpos()`, `__gnu_cxx::stdio_filebuf< _CharT, _Traits >::stdio_filebuf()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, `std::basic_filebuf< _CharT, _Traits >::xsgetn()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

The documentation for this class was generated from the following files:

- [fstream](#)
- [fstream.tcc](#)

5.611 std::basic_fstream<_CharT, _Traits> Class Template Reference

Inheritance diagram for std::basic_fstream<_CharT, _Traits>:



Public Types

- typedef `ctype<_CharT>` `__ctype_type`
- typedef `ctype<_CharT>` `__ctype_type`
- typedef `basic_filebuf<char_type, traits_type>` `__filebuf_type`
- typedef `basic_ios<char_type, traits_type>` `__ios_type`
- typedef `basic_iostream<char_type, traits_type>` `__iostream_type`
- typedef `basic_istream<_CharT, _Traits>` `__istream_type`
- typedef `num_get<_CharT, istreambuf_iterator<_CharT, _Traits>>` `__num_get_type`
- typedef `num_put<_CharT, ostreambuf_iterator<_CharT, _Traits>>` `__num_put_type`
- typedef `basic_ostream<_CharT, _Traits>` `__ostream_type`
- typedef `basic_streambuf<_CharT, _Traits>` `__streambuf_type`
- typedef `basic_streambuf<_CharT, _Traits>` `__streambuf_type`
- typedef `_CharT` `char_type`
- enum `event` { `erase_event`, `imbue_event`, `copyfmt_event` }

- typedef void(* [event_callback](#)) ([event](#) __e, [ios_base](#) & __b, int __i)
 - typedef [_ios_Fmtflags](#) [fmtflags](#)
 - typedef traits_type::int_type [int_type](#)
 - typedef int [io_state](#)
 - typedef [_ios_iostate](#) [iostate](#)
 - typedef traits_type::off_type [off_type](#)
 - typedef int [open_mode](#)
 - typedef [_ios_Openmode](#) [openmode](#)
 - typedef traits_type::pos_type [pos_type](#)
 - typedef int [seek_dir](#)
 - typedef [_ios_Seekdir](#) [seekdir](#)
 - typedef [std::streamoff](#) [streamoff](#)
 - typedef [std::streampos](#) [streampos](#)
 - typedef [_Traits](#) [traits_type](#)
-
- typedef [num_put](#)< [_CharT](#), [ostreambuf_iterator](#)< [_CharT](#), [_Traits](#) > > [__num_put_type](#)

Public Member Functions

- [basic_fstream](#) ()
- [basic_fstream](#) (const char *__s, [ios_base::openmode](#) __mode=[ios_base::in|ios_base::out](#))
- [basic_fstream](#) (const [std::string](#) & __s, [ios_base::openmode](#) __mode=[ios_base::in|ios_base::out](#))
- [basic_fstream](#) (const [basic_fstream](#) &)=delete
- [basic_fstream](#) ([basic_fstream](#) && __rhs)
- [~basic_fstream](#) ()
- template<typename [_ValueT](#) >
 [basic_istream](#)< [_CharT](#), [_Traits](#) > & [_M_extract](#) ([_ValueT](#) & __v)
- const [locale](#) & [_M_getloc](#) () const
- template<typename [_ValueT](#) >
 [basic_ostream](#)< [_CharT](#), [_Traits](#) > & [_M_insert](#) ([_ValueT](#) __v)
- void [_M_setstate](#) ([iostate](#) __state)
- bool [bad](#) () const
- void [clear](#) ([iostate](#) __state=[goodbit](#))
- void [close](#) ()
- [basic_ios](#) & [copyfmt](#) (const [basic_ios](#) & __rhs)
- bool [eof](#) () const
- [iostate](#) [exceptions](#) () const
- void [exceptions](#) ([iostate](#) __except)
- bool [fail](#) () const
- char_type [fill](#) () const
- char_type [fill](#) (char_type __ch)
- [fmtflags](#) [flags](#) () const
- [fmtflags](#) [flags](#) ([fmtflags](#) __fmtfl)
- [__ostream_type](#) & [flush](#) ()
- [streamsize](#) [gcount](#) () const
- template<>
 [basic_istream](#)< char > & [getline](#) (char_type *__s, [streamsize](#) __n, char_type __delim)
- template<>
 [basic_istream](#)< wchar_t > & [getline](#) (char_type *__s, [streamsize](#) __n, char_type __delim)

- [locale getloc](#) () const
 - [bool good](#) () const
 - [template<>](#)
[basic_istream](#)< char > & [ignore](#) (streamsize __n)
 - [template<>](#)
[basic_istream](#)< char > & [ignore](#) (streamsize __n, int_type __delim)
 - [template<>](#)
[basic_istream](#)< wchar_t > & [ignore](#) (streamsize __n)
 - [template<>](#)
[basic_istream](#)< wchar_t > & [ignore](#) (streamsize __n, int_type __delim)
 - [locale imbue](#) (const [locale](#) &__loc)
 - [bool is_open](#) ()
 - [bool is_open](#) () const
 - [long & iword](#) (int __ix)
 - [char narrow](#) (char_type __c, char __dfault) const
 - [void open](#) (const char * __s, [ios_base::openmode](#) __mode=[ios_base::in](#)|[ios_base::out](#))
 - [void open](#) (const [std::string](#) &__s, [ios_base::openmode](#) __mode=[ios_base::in](#)|[ios_base::out](#))
 - [__ostream_type & operator<<](#) (const void * __p)
 - [__ostream_type & operator<<](#) ([__streambuf_type](#) * __sb)
 - [basic_fstream & operator=](#) (const [basic_fstream](#) &)=delete
 - [basic_fstream & operator=](#) ([basic_fstream](#) &&__rhs)
 - [__istream_type & operator>>](#) (void *&__p)
 - [__istream_type & operator>>](#) ([__streambuf_type](#) * __sb)
 - [streamsize precision](#) () const
 - [streamsize precision](#) (streamsize __prec)
 - [void *& pword](#) (int __ix)
 - [basic_streambuf](#)<_CharT, _Traits> * [rdbuf](#) ([basic_streambuf](#)<_CharT, _Traits> * __sb)
 - [__filebuf_type * rdbuf](#) () const
 - [iostate rdstate](#) () const
 - [void register_callback](#) ([event_callback](#) __fn, int __index)
 - [__ostream_type & seekp](#) (pos_type)
 - [__ostream_type & seekp](#) (off_type, [ios_base::seekdir](#))
 - [fmtflags setf](#) (fmtflags __fmtfl)
 - [fmtflags setf](#) (fmtflags __fmtfl, [fmtflags](#) __mask)
 - [void setstate](#) ([iostate](#) __state)
 - [void swap](#) ([basic_fstream](#) &__rhs)
 - [pos_type tellp](#) ()
 - [basic_ostream](#)<_CharT, _Traits> * [tie](#) () const
 - [basic_ostream](#)<_CharT, _Traits> * [tie](#) ([basic_ostream](#)<_CharT, _Traits> * __tiestr)
 - [void unsetf](#) (fmtflags __mask)
 - [char_type widen](#) (char __c) const
 - [streamsize width](#) () const
 - [streamsize width](#) (streamsize __wide)
-
- [__istream_type & operator>>](#) ([__istream_type](#) &(*__pf)([__istream_type](#) &))
 - [__istream_type & operator>>](#) ([__ios_type](#) &(*__pf)([__ios_type](#) &))
 - [__istream_type & operator>>](#) ([ios_base](#) &(*__pf)([ios_base](#) &))

Extractors

All the `operator>>` functions (aka formatted input functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (`noskipws`) set to `false`. This has several effects, concluding with the setting of a status flag; see the `sentry` documentation for more.

If the `sentry` status is good, the function tries to extract whatever data is appropriate for the type of the argument.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- `__istream_type & operator>> (bool &__n)`
- `__istream_type & operator>> (short &__n)`
- `__istream_type & operator>> (unsigned short &__n)`
- `__istream_type & operator>> (int &__n)`
- `__istream_type & operator>> (unsigned int &__n)`
- `__istream_type & operator>> (long &__n)`
- `__istream_type & operator>> (unsigned long &__n)`
- `__istream_type & operator>> (long long &__n)`
- `__istream_type & operator>> (unsigned long long &__n)`
- `__istream_type & operator>> (float &__f)`
- `__istream_type & operator>> (double &__f)`
- `__istream_type & operator>> (long double &__f)`

Unformatted Input Functions

All the unformatted input functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (`noskipws`) set to `true`. This has several effects, concluding with the setting of a status flag; see the `sentry` documentation for more.

If the `sentry` status is good, the function tries to extract whatever data is appropriate for the type of the argument.

The number of characters extracted is stored for later retrieval by `gcount()`.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- `int_type get ()`
- `__istream_type & get (char_type &__c)`
- `__istream_type & get (char_type * __s, streamsize __n, char_type __delim)`
- `__istream_type & get (char_type * __s, streamsize __n)`
- `__istream_type & get (__streambuf_type & __sb, char_type __delim)`
- `__istream_type & get (__streambuf_type & __sb)`
- `__istream_type & getline (char_type * __s, streamsize __n, char_type __delim)`
- `__istream_type & getline (char_type * __s, streamsize __n)`
- `__istream_type & ignore (streamsize __n, int_type __delim)`
- `__istream_type & ignore (streamsize __n)`
- `__istream_type & ignore ()`
- `int_type peek ()`
- `__istream_type & read (char_type * __s, streamsize __n)`
- `streamsize readsome (char_type * __s, streamsize __n)`
- `__istream_type & putback (char_type __c)`
- `__istream_type & unget ()`
- `int sync ()`
- `pos_type tellg ()`
- `__istream_type & seekg (pos_type)`
- `__istream_type & seekg (off_type, ios_base::seekdir)`
- `operator bool () const`

- bool [operator!](#) () const
- [__ostream_type](#) & [operator<<](#) ([__ostream_type](#) &(*__pf)([__ostream_type](#) &))
- [__ostream_type](#) & [operator<<](#) ([__ios_type](#) &(*__pf)([__ios_type](#) &))
- [__ostream_type](#) & [operator<<](#) ([ios_base](#) &(*__pf)([ios_base](#) &))

Inserters

All the `operator<<` functions (aka formatted output functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This can have several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- [__ostream_type](#) & [operator<<](#) (long __n)
- [__ostream_type](#) & [operator<<](#) (unsigned long __n)
- [__ostream_type](#) & [operator<<](#) (bool __n)
- [__ostream_type](#) & [operator<<](#) (short __n)
- [__ostream_type](#) & [operator<<](#) (unsigned short __n)
- [__ostream_type](#) & [operator<<](#) (int __n)
- [__ostream_type](#) & [operator<<](#) (unsigned int __n)
- [__ostream_type](#) & [operator<<](#) (long long __n)
- [__ostream_type](#) & [operator<<](#) (unsigned long long __n)
- [__ostream_type](#) & [operator<<](#) (double __f)
- [__ostream_type](#) & [operator<<](#) (float __f)
- [__ostream_type](#) & [operator<<](#) (long double __f)

Unformatted Output Functions

All the unformatted output functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This has several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state. If `badbit` is on in the stream's exceptions mask, the exception will be rethrown without completing its actions.

- [__ostream_type](#) & [put](#) (char_type __c)
- void [_M_write](#) (const char_type *__s, [streamsize](#) __n)
- [__ostream_type](#) & [write](#) (const char_type *__s, [streamsize](#) __n)

Static Public Member Functions

- static bool [sync_with_stdio](#) (bool __sync=true)
- static int [xalloc](#) () throw ()

Static Public Attributes

- static const [fmtflags adjustfield](#)
- static const [openmode app](#)
- static const [openmode ate](#)
- static const [iosstate badbit](#)
- static const [fmtflags basefield](#)
- static const [seekdir beg](#)
- static const [openmode binary](#)
- static const [fmtflags boolalpha](#)
- static const [seekdir cur](#)
- static const [fmtflags dec](#)
- static const [seekdir end](#)
- static const [iosstate eofbit](#)
- static const [iosstate failbit](#)
- static const [fmtflags fixed](#)
- static const [fmtflags floatfield](#)
- static const [iosstate goodbit](#)
- static const [fmtflags hex](#)
- static const [openmode in](#)
- static const [fmtflags internal](#)
- static const [fmtflags left](#)
- static const [fmtflags oct](#)
- static const [openmode out](#)
- static const [fmtflags right](#)
- static const [fmtflags scientific](#)
- static const [fmtflags showbase](#)
- static const [fmtflags showpoint](#)
- static const [fmtflags showpos](#)
- static const [fmtflags skipws](#)
- static const [openmode trunc](#)
- static const [fmtflags unitbuf](#)
- static const [fmtflags uppercase](#)

Protected Types

- enum { [_S_local_word_size](#) }

Protected Member Functions

- void [_M_cache_locale](#) (const [locale](#) &__loc)
- void [_M_call_callbacks](#) ([event](#) __ev) throw ()
- void [_M_dispose_callbacks](#) (void) throw ()
- template<typename _ValueT >
[__istream_type](#) & [_M_extract](#) (_ValueT &__v)
- [_Words](#) & [_M_grow_words](#) (int __index, bool __iword)
- void [_M_init](#) () throw ()
- template<typename _ValueT >
[__ostream_type](#) & [_M_insert](#) (_ValueT __v)

- void **_M_move** (ios_base &) noexcept
- void **_M_swap** (ios_base &__rhs) noexcept
- void **init** (basic_streambuf<_CharT, _Traits> *__sb)
- void **move** (basic_ios &__rhs)
- void **move** (basic_ios &&__rhs)
- void **set_rdbuf** (basic_streambuf<_CharT, _Traits> *__sb)
- void **swap** (basic_ostream &__rhs)
- void **swap** (basic_ios &__rhs) noexcept
- void **swap** (basic_istream &__rhs)
- void **swap** (basic_iostream &__rhs)

Protected Attributes

- _Callback_list * **_M_callbacks**
- const __ctype_type * **_M_ctype**
- iostate **_M_exception**
- char_type **_M_fill**
- bool **_M_fill_init**
- fmtflags **_M_flags**
- streamsize **_M_gcount**
- locale **_M_ios_locale**
- _Words **_M_local_word** [_S_local_word_size]
- const __num_get_type * **_M_num_get**
- const __num_put_type * **_M_num_put**
- streamsize **_M_precision**
- basic_streambuf<_CharT, _Traits> * **_M_streambuf**
- iostate **_M_streambuf_state**
- basic_ostream<_CharT, _Traits> * **_M_tie**
- streamsize **_M_width**
- _Words * **_M_word**
- int **_M_word_size**
- _Words **_M_word_zero**

5.611.1 Detailed Description

```
template<typename _CharT, typename _Traits>
class std::basic_fstream<_CharT, _Traits>
```

Controlling input and output for files.

Template Parameters

<code>_CharT</code>	Type of character stream.
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .

This class supports reading from and writing to named files, using the inherited functions from `std::basic_iostream`. To control the associated sequence, an instance of `std::basic_filebuf` is used, which this page refers to as `sb`.

Definition at line 862 of file fstream.

5.611.2 Member Typedef Documentation

5.611.2.1 `template<typename _CharT, typename _Traits> typedef num_put<_CharT, ostreambuf_iterator<_CharT, _Traits>
> std::basic_ios<_CharT, _Traits>::__num_put_type [inherited]`

These are non-standard types.

Definition at line 89 of file basic_ios.h.

5.611.2.2 `typedef void(* std::ios_base::event_callback)(event __e, ios_base &__b, int __i) [inherited]`

The type of an event callback function.

Parameters

<code>__e</code>	One of the members of the event enum.
<code>__b</code>	Reference to the ios_base object.
<code>__i</code>	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several ios_base and basic_ios functions, specifically imbue(), copyfmt(), and ~ios().

Definition at line 504 of file ios_base.h.

5.611.2.3 `typedef _Ios_Fmtflags std::ios_base::fmtflags [inherited]`

This is a bitmask type.

`_Ios_Fmtflags` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type fmtflags are:

- boolalpha
- dec
- fixed
- hex
- internal
- left
- oct

- `right`
- `scientific`
- `showbase`
- `showpoint`
- `showpos`
- `skipws`
- `unitbuf`
- `uppercase`
- `adjustfield`
- `basefield`
- `floatfield`

Definition at line 323 of file `ios_base.h`.

5.611.2.4 `typedef _Ios_Iostate std::ios_base::iostate` [inherited]

This is a bitmask type.

`_Ios_Iostate` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `iostate` are:

- `badbit`
- `eofbit`
- `failbit`
- `goodbit`

Definition at line 398 of file `ios_base.h`.

5.611.2.5 `typedef _Ios_Openmode std::ios_base::openmode` [inherited]

This is a bitmask type.

`_Ios_Openmode` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `openmode` are:

- `app`
- `ate`
- `binary`
- `in`
- `out`
- `trunc`

Definition at line 429 of file `ios_base.h`.

5.611.2.6 `typedef _Ios_Seekdir std::ios_base::seekdir` [inherited]

This is an enumerated type.

`_Ios_Seekdir` is implementation-defined. Defined values of type `seekdir` are:

- `beg`
- `cur`, equivalent to `SEEK_CUR` in the C standard library.
- `end`, equivalent to `SEEK_END` in the C standard library.

Definition at line 461 of file `ios_base.h`.

5.611.3 Member Enumeration Documentation

5.611.3.1 `enum std::ios_base::event` [inherited]

The set of events that may be passed to an event callback.

`erase_event` is used during `~ios()` and `copyfmt()`. `imbue_event` is used during `imbue()`. `copyfmt_event` is used during `copyfmt()`.

Definition at line 487 of file `ios_base.h`.

5.611.4 Constructor & Destructor Documentation

5.611.4.1 `template<typename _CharT, typename _Traits> std::basic_fstream<_CharT, _Traits>::basic_fstream ()` [inline]

Default constructor.

Initializes `sb` using its default constructor, and passes `&sb` to the base class initializer. Does not open any files (you haven't given it a filename to open).

Definition at line 889 of file `fstream`.

5.611.4.2 `template<typename _CharT, typename _Traits> std::basic_fstream<_CharT, _Traits>::basic_fstream (const char * __s, ios_base::openmode __mode = ios_base::in | ios_base::out)` [inline], [explicit]

Create an input/output file stream.

Parameters

<code>__s</code>	Null terminated string specifying the filename.
<code>__mode</code>	Open file in specified mode (see <code>std::ios_base</code>).

Tip: When using std::string to hold the filename, you must use .c_str() before passing it to this constructor.

Definition at line 902 of file fstream.

```
5.611.4.3 template<typename _CharT, typename _Traits> std::basic_fstream<_CharT, _Traits>::basic_fstream (
    const std::string & __s, ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline],
    [explicit]
```

Create an input/output file stream.

Parameters

<code>__s</code>	Null terminated string specifying the filename.
<code>__mode</code>	Open file in specified mode (see std::ios_base).

Definition at line 917 of file fstream.

```
5.611.4.4 template<typename _CharT, typename _Traits> std::basic_fstream<_CharT, _Traits>::~~basic_fstream ( )
    [inline]
```

The destructor does nothing.

The file is closed by the filebuf object, not the formatting stream.

Definition at line 939 of file fstream.

5.611.5 Member Function Documentation

```
5.611.5.1 const locale& std::ios_base::_M_getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A reference to the current locale.

Like getloc above, but returns a reference instead of generating a copy.

Definition at line 774 of file ios_base.h.

Referenced by std::time_get<_CharT, _Inlter>::do_date_order(), std::time_get<_CharT, _Inlter>::do_get(), std::money_get<_CharT, _Inlter>::do_get(), std::num_get<_CharT, _Inlter>::do_get(), std::time_get<_CharT, _Inlter>::do_get_date(), std::time_get<_CharT, _Inlter>::do_get_monthname(), std::time_get<_CharT, _Inlter>::do_get_time(), std::time_get<_CharT, _Inlter>::do_get_weekday(), std::time_get<_CharT, _Inlter>::do_get_year(), std::time_put<_CharT, _Outlter>::do_put(), std::num_put<_CharT, _Outlter>::do_put(), std::time_get<_CharT, _Inlter>::get(), and std::time_put<_CharT, _Outlter>::put().

```
5.611.5.2 template<typename _CharT, typename _Traits> void std::basic_ostream<_CharT, _Traits>::_M_write ( const
    char_type * __s, streamsize __n ) [inline], [inherited]
```

Core write functionality, without sentry.

Parameters

<code>_↵ _s</code>	The array to insert.
<code>_↵ _n</code>	Maximum number of characters to insert.

Definition at line 311 of file ostream.

Referenced by `std::basic_ostream<_CharT, _Traits>::write()`.

5.611.5.3 `template<typename _CharT, typename _Traits> bool std::basic_ios<_CharT, _Traits>::bad () const`
`[inline], [inherited]`

Fast error checking.

Returns

True if the badbit is set.

Note that other iostate flags may also be set.

Definition at line 211 of file basic_ios.h.

5.611.5.4 `template<typename _CharT, typename _Traits> void std::basic_ios<_CharT, _Traits>::clear (iostate __state =`
`goodbit) [inherited]`

[Re]sets the error state.

Parameters

<code>__state</code>	The new state flag(s) to set.
----------------------	-------------------------------

See `std::ios_base::iostate` for the possible bit values. Most users will not need to pass an argument.

Definition at line 41 of file basic_ios.tcc.

Referenced by `std::basic_ios<char, _Traits>::exceptions()`, `std::__detail::operator>>()`, `std::basic_istream<_CharT, _Traits>::putback()`, `std::basic_ios<char, _Traits>::rdstate()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_ios<char, _Traits>::setstate()`, and `std::basic_istream<_CharT, _Traits>::unget()`.

5.611.5.5 `template<typename _CharT, typename _Traits> void std::basic_fstream<_CharT, _Traits>::close ()`
`[inline]`

Close the file.

Calls `std::basic_filebuf::close()`. If that function fails, `failbit` is set in the stream's error state.

Definition at line 1041 of file fstream.

5.611.5.6 `template<typename _CharT, typename _Traits> basic_ios<_CharT, _Traits> & std::basic_ios<_CharT, _Traits>::copyfmt(const basic_ios<_CharT, _Traits> & __rhs) [inherited]`

Copies fields of `__rhs` into this.

Parameters

<code>__rhs</code>	The source values for the copies.
--------------------	-----------------------------------

Returns

Reference to this object.

All fields of `__rhs` are copied into this object except that `rdbuf()` and `rdstate()` remain unchanged. All values in the `pword` and `iword` arrays are copied. Before copying, each callback is invoked with `erase_event`. After copying, each (new) callback is invoked with `copyfmt_event`. The final step is to copy exceptions().

Definition at line 63 of file `basic_ios.tcc`.

References `std::basic_ios<_CharT, _Traits>::exceptions()`, `std::basic_ios<_CharT, _Traits>::fill()`, `std::ios_base::flags()`, `std::ios_base::getloc()`, `std::ios_base::precision()`, `std::basic_ios<_CharT, _Traits>::tie()`, `std::tie()`, and `std::ios_base::width()`.

Referenced by `std::basic_ios<char, _Traits>::rdbuf()`.

5.611.5.7 `template<typename _CharT, typename _Traits> bool std::basic_ios<_CharT, _Traits>::eof() const [inline], [inherited]`

Fast error checking.

Returns

True if the eofbit is set.

Note that other iostate flags may also be set.

Definition at line 190 of file `basic_ios.h`.

5.611.5.8 `template<typename _CharT, typename _Traits> iostate std::basic_ios<_CharT, _Traits>::exceptions() const [inline], [inherited]`

Throwing exceptions on errors.

Returns

The current exceptions mask.

This changes nothing in the stream. See the one-argument version of `exceptions(iostate)` for the meaning of the return value.

Definition at line 222 of file `basic_ios.h`.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, and `std::basic_ios<char, _Traits>::setstate()`.

5.611.5.9 `template<typename _CharT, typename _Traits> void std::basic_ios<_CharT, _Traits>::exceptions(iostate __except) [inline], [inherited]`

Throwing exceptions on errors.

Parameters

<code>__except</code>	The new exceptions mask.
-----------------------	--------------------------

By default, error flags are set silently. You can set an exceptions mask for each stream; if a bit in the mask becomes set in the error flags, then an exception of type `std::ios_base::failure` is thrown.

If the error flag is already set when the exceptions mask is added, the exception is immediately thrown. Try running the following under GCC 3.1 or later:

```

1 #include <iostream>
2 #include <fstream>
3 #include <exception>
4
5 int main()
6 {
7     std::set_terminate (__gnu_cxx::__verbose_terminate_handler);
8
9     std::ifstream f ("/etc/motd");
10
11     std::cerr << "Setting badbit\n";
12     f.setstate (std::ios_base::badbit);
13
14     std::cerr << "Setting exception mask\n";
15     f.exceptions (std::ios_base::badbit);
16 }
```

Definition at line 257 of file `basic_ios.h`.

5.611.5.10 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::fail () const`
`[inline], [inherited]`

Fast error checking.

Returns

True if either the badbit or the failbit is set.

Checking the badbit in `fail()` is historical practice. Note that other iostate flags may also be set.

Definition at line 201 of file `basic_ios.h`.

Referenced by `std::basic_ios< char, _Traits >::operator bool()`, `std::basic_ios< char, _Traits >::operator!()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, and `std::regex_traits< _Ch_type >::value()`.

5.611.5.11 `template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::fill () const`
`[inline], [inherited]`

Retrieves the *empty* character.

Returns

The current fill character.

It defaults to a space (' ') in the current locale.

Definition at line 370 of file `basic_ios.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::basic_ios< char, _Traits >::fill()`, `std::normal_distribution< _RealType >::operator()()`, `std::gamma_distribution< _RealType >::operator()()`, `std::negative_binomial_distribution< _IntType >::operator()()`, and `std::operator>>()`.

5.611.5.12 `template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::fill (char_type __ch) [inline], [inherited]`

Sets a new *empty* character.

Parameters

<code>__ch</code>	The new character.
-------------------	--------------------

Returns

The previous fill character.

The fill character is used to fill out space when P+ characters have been requested (e.g., via `setw`), Q characters are actually used, and Q<P. It defaults to a space (' ') in the current locale.

Definition at line 390 of file `basic_ios.h`.

5.611.5.13 `fmtflags std::ios_base::flags () const [inline], [inherited]`

Access to format flags.

Returns

The format control flags for both input and output.

Definition at line 619 of file `ios_base.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >::discard()`, `std::money_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::operator()()`, `std::discard_block_engine< _RandomNumberEngine, __p, __r >::operator()()`, `std::shuffle_order_engine< _RandomNumberEngine, __k >::operator()()`, `std::normal_distribution< _RealType >::operator()()`, `std::gamma_distribution< _RealType >::operator()()`, `std::binomial_distribution< _IntType >::operator()()`, `std::negative_binomial_distribution< _IntType >::operator()()`, `std::poisson_distribution< _IntType >::operator()()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, `std::__detail::operator>>()`, `std::operator>>()`, `std::linear_congruential_engine< _UIntType, __a, __c, __m >::seed()`, and `std::basic_istream< _CharT, _Traits >::sentry::sentry()`.

5.611.5.14 `fmtflags std::ios_base::flags (fmtflags __fmtfl) [inline], [inherited]`

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

Definition at line 630 of file `ios_base.h`.

```
5.611.5.15  template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits> & std::basic_ostream<
            _CharT, _Traits>::flush( ) [inherited]
```

Synchronizing the stream buffer.

Returns

*this

If `rdbuf()` is a null pointer, changes nothing.

Otherwise, calls `rdbuf()->pubsync()`, and if that returns -1, sets `badbit`.

Definition at line 211 of file `ostream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::setstate()`, and `std::basic_ostream<_CharT, _Traits>::tellp()`.

Referenced by `std::operator<<()`, and `std::basic_ostream<_CharT, _Traits>::write()`.

```
5.611.5.16  template<typename _CharT, typename _Traits> streamsize std::basic_istream<_CharT, _Traits>::gcount( )
            const [inline],[inherited]
```

Character counting.

Returns

The number of characters extracted by the previous unformatted input function dispatched for this stream.

Definition at line 269 of file `istream`.

```
5.611.5.17  template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits>::int_type
            std::basic_istream<_CharT, _Traits>::get( void ) [inherited]
```

Simple extraction.

Returns

A character, or `eof()`.

Tries to extract a character. If none are available, sets `failbit` and returns `traits::eof()`.

Definition at line 236 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream<_CharT, _Traits>::get()`, and `std::basic_istream<_CharT, _Traits>::operator>>()`.

```
5.611.5.18  template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<
            _CharT, _Traits>::get( char_type & __c ) [inherited]
```

Simple extraction.

Parameters

<code>_↵</code>	The character in which to store data.
<code>_c</code>	

Returns

*this

Tries to extract a character and store it in `__c`. If none are available, sets failbit and returns `traits::eof()`.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 272 of file `istream.tcc`.

References `std::basic_istream< _CharT, _Traits >::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_↵`
`base::failbit`, `std::basic_istream< _CharT, _Traits >::get()`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >↵`
`::rdbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

5.611.5.19 `template<typename _CharT, typename _Traits > basic_istream< _CharT, _Traits > & std::basic_istream<`
`_CharT, _Traits >::get (char_type * __s, streamsize __n, char_type __delim)` [inherited]

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
<code>__n</code>	Maximum number of characters to store in <code>__s</code> .
<code>__delim</code>	A "stop" character.

Returns

*this

Characters are extracted and stored into `__s` until one of the following happens:

- `__n-1` characters are stored
- the input sequence reaches EOF
- the next character equals `__delim`, in which case the character is not extracted

If no characters are stored, failbit is set in the stream's error state.

In any case, a null character is stored into the next location in the array.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 309 of file istream.tcc.

References `std::basic_istream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::basic_istream<_CharT, _Traits>::get()`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sgetc()`, and `std::basic_streambuf<_CharT, _Traits>::snextc()`.

5.611.5.20 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::get (char_type* __s, streamsize __n) [inline], [inherited]`

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
<code>__n</code>	Maximum number of characters to store in <code>s</code> .

Returns

`*this`

Returns `get(__s, __n, widen("\n"))`.

Definition at line 354 of file istream.

5.611.5.21 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::get (__streambuf_type & __sb, char_type __delim) [inherited]`

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
<code>__delim</code>	A "stop" character.

Returns

`*this`

Characters are extracted and inserted into `__sb` until one of the following happens:

- the input sequence reaches EOF

- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted)
- the next character equals `__delim` (in this case, the character is not extracted)
- an exception occurs (and in this case is caught)

If no characters are stored, failbit is set in the stream's error state.

Definition at line 356 of file istream.tcc.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_ios< _CharT, _Traits >::setstate()`, `std::basic_streambuf< _CharT, _Traits >::sgetc()`, `std::basic_streambuf< _CharT, _Traits >::snextc()`, and `std::basic_streambuf< _CharT, _Traits >::sputc()`.

5.611.5.22 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits >::get (__streambuf_type & __sb)` `[inline]`, `[inherited]`

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
-------------------	-------------------------------------

Returns

`*this`

Returns `get(__sb,widen("\n"))`.

Definition at line 387 of file istream.

5.611.5.23 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::getline (char_type * __s, streamsize __n, char_type __delim)` `[inherited]`

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.
<code>__delim</code>	A "stop" character.

Returns

`*this`

Extracts and stores characters into `__s` until one of the following happens. Note that these criteria are required to be tested in the order listed here, to allow an input line to exactly fill the `__s` array without setting failbit.

1. the input sequence reaches end-of-file, in which case eofbit is set in the stream error state
2. the next character equals `__delim`, in which case the character is extracted (and therefore counted in `gcount()`) but not stored
3. `__n-1` characters are stored, in which case failbit is set in the stream error state

If no characters are extracted, failbit is set. (An empty line of input should therefore not cause failbit to be set.)

In any case, a null character is stored in the next location in the array.

Definition at line 400 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_streambuf<_CharT, _Traits>::sbumpc()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sgetc()`, and `std::basic_streambuf<_CharT, _Traits>::snextc()`.

Referenced by `std::basic_istream<_CharT, _Traits>::get()`.

5.611.5.24 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::getline (char_type * __s, streamsize __n) [inline], [inherited]`

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.

Returns

`*this`

Returns `getline(__s, __n, widen("\n"))`.

Definition at line 427 of file `istream`.

5.611.5.25 `template<> basic_istream<char> & std::basic_istream<char>::getline (char_type * __s, streamsize __n, char_type __delim) [inherited]`

Explicit specialization declarations, defined in `src/istream.cc`.

5.611.5.26 `locale std::ios_base::getloc () const [inline], [inherited]`

Locale access.

Returns

A copy of the current locale.

If `imbue(loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale()`, the global C++ locale.

Definition at line 763 of file `ios_base.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::money_put< _CharT, _OutIter >::do_put()`, `std::operator>>()`, and `std::ws()`.

5.611.5.27 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::good () const`
`[inline], [inherited]`

Fast error checking.

Returns

True if no error flags are set.

A wrapper around `rdstate`.

Definition at line 180 of file `basic_ios.h`.

Referenced by `std::__detail::operator>>()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_istream< _CharT, _Traits >::sentry::sentry()`.

5.611.5.28 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::ignore (streamsize __n, int_type __delim)` `[inherited]`

Discarding characters.

Parameters

<code>__n</code>	Number of characters to discard.
<code>__delim</code>	A "stop" character.

Returns

`*this`

Extracts characters and throws them away until one of the following happens:

- if `__n != std::numeric_limits<int>::max()`, `__n` characters are extracted
- the input sequence reaches end-of-file
- the next character equals `__delim` (in this case, the character is extracted); note that this condition will never occur if `__delim` equals `traits::eof()`.

NB: Provide three overloads, instead of the single function (with defaults) mandated by the Standard: this leads to a better performing implementation, while still conforming to the Standard.

Definition at line 555 of file istream.tcc.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_istreambuf< _CharT, _Traits >::sbumpc()`, `std::basic_ios< _CharT, _Traits >::setstate()`, `std::basic_istreambuf< _CharT, _Traits >::sgetc()`, and `std::basic_istreambuf< _CharT, _Traits >::snextc()`.

5.611.5.29 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::ignore (streamsize __n) [inherited]`

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns `traits::eof()`.

Definition at line 493 of file istream.tcc.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_ios< _CharT, _Traits >::setstate()`, `std::basic_istreambuf< _CharT, _Traits >::sgetc()`, and `std::basic_istreambuf< _CharT, _Traits >::snextc()`.

5.611.5.30 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::ignore (void) [inherited]`

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns `traits::eof()`.

Definition at line 460 of file istream.tcc.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_istreambuf< _CharT, _Traits >::sbumpc()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

Referenced by `std::basic_istream< _CharT, _Traits >::getline()`, and `std::basic_istream< _CharT, _Traits >::ignore()`.

5.611.5.31 `template<typename _CharT, typename _Traits> locale std::basic_ios< _CharT, _Traits>::imbue (const locale & __loc) [inherited]`

Moves to a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls `ios_base::imbue(loc)`, and if a stream buffer is associated with this stream, calls that buffer's `pubimbue(loc)`.

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>.

Definition at line 114 of file `basic_ios.tcc`.

References `std::ios_base::imbue()`.

Referenced by `std::basic_ios<char, _Traits>::fill()`.

5.611.5.32 `template<typename _CharT, typename _Traits> void std::basic_ios<_CharT, _Traits>::init(basic_streambuf<_CharT, _Traits> * __sb)` `[protected]`, `[inherited]`

All setup is performed here.

This is called from the public constructor. It is not virtual and cannot be redefined.

Definition at line 126 of file `basic_ios.tcc`.

Referenced by `std::basic_ios<char, _Traits>::basic_ios()`.

5.611.5.33 `template<typename _CharT, typename _Traits> bool std::basic_fstream<_CharT, _Traits>::is_open()` `[inline]`

Wrapper to test for an open file.

Returns

`rdbuf() -> is_open()`

Definition at line 980 of file `fstream`.

5.611.5.34 `long& std::ios_base::iword(int __ix)` `[inline]`, `[inherited]`

Access to integer array.

Parameters

<code>_↔ _ix</code>	Index into the array.
-------------------------	-----------------------

Returns

A reference to an integer associated with the index.

The `iword` function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 809 of file `ios_base.h`.

```
5.611.5.35  template<typename _CharT, typename _Traits> char std::basic_ios< _CharT, _Traits >::narrow ( char_type __c,
char __default ) const    [inline], [inherited]
```

Squeezes characters.

Parameters

<code>__c</code>	The character to narrow.
<code>__default</code>	The character to narrow.

Returns

The narrowed character.

Maps a character of `char_type` to a character of `char`, if possible.

Returns the result of

```
1  std::use_facet<ctype<char_type>> >(getloc()).narrow(c,default)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.↔html>

Definition at line 430 of file `basic_ios.h`.

```
5.611.5.36  template<typename _CharT, typename _Traits> void std::basic_fstream< _CharT, _Traits >::open ( const char *
__s, ios_base::openmode __mode = ios_base::in | ios_base::out )    [inline]
```

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Calls `std::basic_filebuf::open(__s, __mode)`. If that function fails, `failbit` is set in the stream's error state.

Tip: When using `std::string` to hold the filename, you must use `.c_str()` before passing it to this constructor.

Definition at line 1001 of file `fstream`.

```
5.611.5.37 template<typename _CharT, typename _Traits> void std::basic_fstream<_CharT, _Traits>::open ( const
std::string & __s, ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline]
```

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Calls `std::basic_filebuf::open(__s, __mode)`. If that function fails, `failbit` is set in the stream's error state.

Definition at line 1022 of file `fstream`.

```
5.611.5.38 template<typename _CharT, typename _Traits> std::basic_ios<_CharT, _Traits>::operator bool ( ) const
[inline], [explicit], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`

Definition at line 117 of file `basic_ios.h`.

```
5.611.5.39 template<typename _CharT, typename _Traits> bool std::basic_ios<_CharT, _Traits>::operator! ( ) const
[inline], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`

Definition at line 125 of file `basic_ios.h`.

5.611.5.40 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits >::operator<<(__ostream_type &(*)(__ostream_type &)__pf) [inline], [inherited]`

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

Definition at line 108 of file `ostream`.

Referenced by `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, and `std::basic_ostream< ↵ _CharT, _Traits >::sentry::sentry()`.

5.611.5.41 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits >::operator<<(__ios_type &(*)(__ios_type &)__pf) [inline], [inherited]`

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

Definition at line 117 of file `ostream`.

5.611.5.42 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits >::operator<<(ios_base &(*)(ios_base &)__pf) [inline], [inherited]`

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

Definition at line 127 of file `ostream`.

5.611.5.43 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits >::operator<<(long __n) [inline], [inherited]`

Integer arithmetic inserters.

Parameters

<code>↵</code>	A variable of builtin integral type.
<code>__n</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 166 of file `ostream`.

5.611.5.44 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(unsigned long __n) [inline], [inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 170 of file `ostream`.

5.611.5.45 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(bool __n) [inline], [inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 174 of file `ostream`.

5.611.5.46 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::operator<<(short __n) [inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 92 of file `ostream.tcc`.

References `std::ios_base::basefield`, `std::ios_base::flags()`, `std::ios_base::hex`, `std::ios_base::oct`, and `std::basic_ostream<_CharT, _Traits>::operator<<()`.

5.611.5.47 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(unsigned short __n) [inline], [inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 181 of file `ostream`.

5.611.5.48 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::operator<<(int __n) [inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 106 of file `ostream.tcc`.

References `std::ios_base::basefield`, `std::ios_base::flags()`, `std::ios_base::hex`, `std::ios_base::oct`, and `std::basic_ostream<_CharT, _Traits>::operator<<()`.

5.611.5.49 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(unsigned int __n) [inline],[inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 192 of file `ostream`.

5.611.5.50 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(long long __n) [inline],[inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 201 of file `ostream`.

5.611.5.51 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(unsigned long long __n) [inline],[inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 205 of file `ostream`.

```
5.611.5.52  template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits
              >::operator<<( double __f )  [inline], [inherited]
```

Floating point arithmetic inserters.

Parameters

↵	A variable of builtin floating point type.
↵	
↵	
↵	
<i>f</i>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 220 of file `ostream`.

```
5.611.5.53  template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits
              >::operator<<( float __f )  [inline], [inherited]
```

Floating point arithmetic inserters.

Parameters

↵	A variable of builtin floating point type.
↵	
↵	
↵	
<i>f</i>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 224 of file `ostream`.

5.611.554 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(long double __f) [inline], [inherited]`

Floating point arithmetic inserters.

Parameters

<code>↔</code>	A variable of builtin floating point type.
<code>__↔</code>	
<code>↔</code>	
<code>__↔</code>	
<code>f</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 232 of file `ostream`.

5.611.555 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(const void * __p) [inline], [inherited]`

Pointer arithmetic inserters.

Parameters

<code>__↔</code>	A variable of pointer type.
<code>__p</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 245 of file `ostream`.

5.611.556 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::operator<<(__streambuf_type * __sb) [inherited]`

Extracting from another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is NULL, the stream will set failbit in its error state.

Characters are extracted from `__sb` and inserted into `*this` until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output sequence fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs while getting a character from `__sb`, which sets failbit in the error state

If the function inserts no characters, failbit is set.

Definition at line 120 of file `ostream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ostream<_CharT, _Traits>::put()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

5.611.5.57 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (__istream_type &(*)(__istream_type &)_pf) [inline],[inherited]`

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

Definition at line 120 of file `istream`.

Referenced by `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::basic_istream<_CharT, _Traits>::sentry←::sentry()`, and `std::ws()`.

5.611.5.58 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (__ios_type &(*)(__ios_type &)_pf) [inline],[inherited]`

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

Definition at line 124 of file `istream`.

5.611.5.59 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (ios_base &(*)(ios_base &)_pf) [inline],[inherited]`

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

Definition at line 131 of file `istream`.

5.611.5.60 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (bool &_n) [inline],[inherited]`

Integer arithmetic extractors.

Parameters

<code>__↔ __n</code>	A variable of builtin integral type.
--------------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 168 of file `istream`.

```
5.611.5.61 template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::operator>> ( short & __n ) [inherited]
```

Integer arithmetic extractors.

Parameters

<code>__↔ __n</code>	A variable of builtin integral type.
--------------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 114 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::failbit`, `std::num_get<_CharT, _InIter>::get()`, `std::ios_base::goodbit`, `std::basic_istream<_CharT, _Traits>::operator>>()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

```
5.611.5.62 template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> ( unsigned short & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

<code>__↔ __n</code>	A variable of builtin integral type.
--------------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 175 of file `istream`.

5.611.5.63 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::operator>> (int & __n) [inherited]`

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 159 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::failbit`, `std::num_get<_CharT, _InIter>::get()`, `std::ios_base::goodbit`, `std::basic_istream<_CharT, _Traits>::operator>>()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

5.611.5.64 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (unsigned int & __n) [inline], [inherited]`

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 182 of file `istream`.

```
5.611.5.65 template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> ( long & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

<code>_↔</code>	A variable of builtin integral type.
<code>_n</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 186 of file `istream`.

```
5.611.5.66  template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits
              >::operator>> ( unsigned long & _n )  [inline],[inherited]
```

Integer arithmetic extractors.

Parameters

<code>_↔</code>	A variable of builtin integral type.
<code>_n</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 190 of file `istream`.

```
5.611.5.67  template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits
              >::operator>> ( long long & _n )  [inline],[inherited]
```

Integer arithmetic extractors.

Parameters

<code>_↔</code>	A variable of builtin integral type.
<code>_n</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 195 of file `istream`.

5.611.5.68 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (unsigned long long &__n)` [inline], [inherited]

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 199 of file `istream`.

5.611.5.69 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (float &__f)` [inline], [inherited]

Floating point arithmetic extractors.

Parameters

<code>__f</code>	A variable of builtin floating point type.
------------------	--

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 214 of file `istream`.

5.611.5.70 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (double &__f)` [inline], [inherited]

Floating point arithmetic extractors.

Parameters

<code>↔</code>	A variable of builtin floating point type.
<code>↔</code>	
<code>↔</code>	
<code>↔</code>	
<code>f</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 218 of file `istream`.

5.611.5.71 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::operator>> (long double &__f) [inline], [inherited]`

Floating point arithmetic extractors.

Parameters

<code>↔</code>	A variable of builtin floating point type.
<code>↔</code>	
<code>↔</code>	
<code>↔</code>	
<code>f</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 222 of file `istream`.

5.611.5.72 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::operator>> (void *&__p) [inline], [inherited]`

Basic arithmetic extractors.

Parameters

<code>↔</code>	A variable of pointer type.
<code>p</code>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 235 of file `istream`.

5.611.5.73 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::operator>> (__streambuf_type * __sb)` `[inherited]`

Extracting into another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is NULL, the stream will set failbit in its error state.

Characters are extracted from this stream and inserted into the `__sb` streambuf until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs (and in this case is caught)

If the function inserts no characters, failbit is set.

Definition at line 204 of file `istream.tcc`.

References `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::basic_istream< _CharT, _Traits>::get()`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

5.611.5.74 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits>::int_type std::basic_istream< _CharT, _Traits>::peek (void)` `[inherited]`

Looking ahead in the stream.

Returns

The next character, or `eof()`.

If, after constructing the sentry object, `good()` is false, returns `traits::eof()`. Otherwise reads but does not extract the next input character.

Definition at line 620 of file `istream.tcc`.

References `std::basic_istream< _CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, `std::basic_istream< _CharT, _Traits>::read()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream< _CharT, _Traits>::ignore()`.

5.611.5.75 `streamsize std::ios_base::precision () const` `[inline],[inherited]`

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Definition at line 689 of file `ios_base.h`.

Referenced by `std::basic_ios<_CharT,_Traits>::copyfmt()`, `std::num_get<_CharT,_InIter>::do_get()`, `std::normal_distribution<_RealType>::operator()()`, `std::gamma_distribution<_RealType>::operator()()`, `std::negative_binomial_distribution<_IntType>::operator()()`, and `std::operator>>()`.

5.611.5.76 `streamsize std::ios_base::precision (streamsize __prec)` `[inline],[inherited]`

Changing flags.

Parameters

<code>__prec</code>	The new precision value.
---------------------	--------------------------

Returns

The previous value of `precision()`.

Definition at line 698 of file `ios_base.h`.

5.611.5.77 `template<typename _CharT,typename _Traits> basic_ostream<_CharT,_Traits> & std::basic_ostream<_CharT,_Traits>::put (char_type __c)` `[inherited]`

Simple insertion.

Parameters

<code>__c</code>	The character to insert.
------------------	--------------------------

Returns

`*this`

Tries to insert `__c`.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 149 of file ostream.tcc.

References std::ios_base::badbit, std::ios_base::goodbit, std::basic_ios< _CharT, _Traits >::rdbuf(), std::basic_ios< _CharT, _Traits >::setstate(), and std::basic_ostream< _CharT, _Traits >::write().

Referenced by std::basic_ostream< _CharT, _Traits >::operator<<().

5.611.5.78 template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::putback(char_type __c) [inherited]

Unextracting a single character.

Parameters

<code>__c</code>	The character to push back into the input stream.
------------------	---

Returns

*this

If rdbuf() is not null, calls rdbuf() -> sputbackc(c).

If rdbuf() is null or if sputbackc() fails, sets badbit in the error state.

Note

This function first clears eofbit. Since no characters are extracted, the next call to gcount() will return 0, as required by DR 60.

Definition at line 711 of file istream.tcc.

References std::basic_istream< _CharT, _Traits >::M_gcount, std::ios_base::badbit, std::basic_ios< _CharT, _Traits >::clear(), std::ios_base::eofbit, std::ios_base::goodbit, std::basic_ios< _CharT, _Traits >::rdbuf(), std::basic_ios< _CharT, _Traits >::rdstate(), std::basic_ios< _CharT, _Traits >::setstate(), std::basic_streambuf< _CharT, _Traits >::sputbackc(), and std::basic_istream< _CharT, _Traits >::unget().

Referenced by std::basic_istream< _CharT, _Traits >::readsome().

5.611.5.79 void*& std::ios_base::pword(int __ix) [inline], [inherited]

Access to void pointer array.

Parameters

<code>_↔ _ix</code>	Index into the array.
-------------------------	-----------------------

Returns

A reference to a `void*` associated with the index.

The `pwdow` function provides access to an array of pointers that can be used for any purpose. The array grows as required to hold the supplied index. All pointers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 830 of file `ios_base.h`.

5.611.5.80 `template<typename _CharT, typename _Traits> basic_streambuf<_CharT, _Traits> * std::basic_ios<_CharT, _Traits>::rdbuf (basic_streambuf<_CharT, _Traits> * __sb) [inherited]`

Changing the underlying buffer.

Parameters

<code>__sb</code>	The new stream buffer.
-------------------	------------------------

Returns

The previous stream buffer.

Associates a new buffer with the current stream, and clears the error state.

Due to historical accidents which the LWG refuses to correct, the I/O library suffers from a design error: this function is hidden in derived classes by overrides of the zero-argument `rdbuf()`, which is non-virtual for hysterical raisins. As a result, you must use explicit qualifications to access this function via any derived class. For example:

```
1 std::fstream    foo;           // or some other derived type
2 std::streambuf* p = .....;
3
4 foo.ios::rdbuf(p);             // ios == basic_ios<char>
```

Definition at line 53 of file `basic_ios.tcc`.

5.611.5.81 `template<typename _CharT, typename _Traits> __filebuf_type* std::basic_fstream<_CharT, _Traits>::rdbuf () const [inline]`

Accessing the underlying buffer.

Returns

The current `basic_filebuf` buffer.

This hides both signatures of `std::basic_ios::rdbuf()`.

Definition at line 972 of file `fstream`.

5.611.5.82 `template<typename _CharT, typename _Traits> istate std::basic_ios< _CharT, _Traits >::rdstate () const`
`[inline], [inherited]`

Returns the error state of the stream buffer.

Returns

A bit pattern (well, isn't everything?)

See `std::ios_base::iostate` for the possible bit values. Most users will call one of the interpreting wrappers, e.g., `good()`.

Definition at line 137 of file `basic_ios.h`.

Referenced by `std::basic_ios< char, _Traits >::bad()`, `std::basic_ios< char, _Traits >::eof()`, `std::basic_ios< char, _Traits >::fail()`, `std::basic_ios< char, _Traits >::good()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ios< char, _Traits >::setstate()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

5.611.5.83 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::read (char_type * __s, streamsize __n)` `[inherited]`

Extraction without delimiters.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

`*this`

If the stream state is `good()`, extracts characters and stores them into `__s` until one of the following happens:

- `__n` characters are stored
- the input sequence reaches end-of-file, in which case the error state is set to `failbit|eofbit`.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 650 of file `istream.tcc`.

References `std::basic_istream< _CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, `std::basic_istream< _CharT, _Traits>::readsome()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream< _CharT, _Traits>::peek()`.

5.611.5.84 `template<typename _CharT, typename _Traits> streamsize std::basic_istream< _CharT, _Traits>::readsome (char_type* __s, streamsize __n) [inherited]`

Extraction until the buffer is exhausted, but no more.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

The number of characters extracted.

Extracts characters and stores them into `__s` depending on the number of characters remaining in the streambuf's buffer, `rdbuf()->in_avail()`, called A here:

- if `A == -1`, sets eofbit and extracts no characters
- if `A == 0`, extracts no characters
- if `A > 0`, extracts `min(A, n)`

The goal is to empty the current buffer, and to not request any more from the external input sequence controlled by the streambuf.

Definition at line 679 of file istream.tcc.

References `std::basic_istream< _CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::min()`, `std::basic_istream< _CharT, _Traits>::putback()`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream< _CharT, _Traits>::read()`.

5.611.5.85 `void std::ios_base::register_callback (event_callback __fn, int __index) [inherited]`

Add the callback `__fn` with parameter `__index`.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

5.611.5.86 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::seekg (pos_type __pos) [inherited]`

Changing the current read position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

`*this`

If `fail()` is not true, calls `rdbuf()->pubseekpos(__pos)`. If that function fails, sets failbit.

Note

This function first clears eofbit. It does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

Definition at line 845 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::ios_base::in`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::rdstate()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream<_CharT, _Traits>::tellg()`.

5.611.5.87 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::seekg (off_type __off, ios_base::seekdir __dir) [inherited]`

Changing the current read position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

`*this`

If `fail()` is not true, calls `rdbuf()->pubseekoff(__off, __dir)`. If that function fails, sets failbit.

Note

This function first clears eofbit. It does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

Definition at line 884 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::ios_base::in`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::rdstate()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

5.611.5.88 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::seekp (pos_type __pos) [inherited]`

Changing the current write position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

`*this`

If `fail()` is not true, calls `rdbuf()->pubseekpos(pos)`. If that function fails, sets failbit.

Definition at line 258 of file `ostream.tcc`.

References `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::ios_base::out`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

Referenced by `std::basic_ostream<_CharT, _Traits>::tellp()`.

5.611.5.89 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::seekp (off_type __off, ios_base::seekdir __dir) [inherited]`

Changing the current write position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

`*this`

If `fail()` is not true, calls `rdbuf()->pubseekoff(off, dir)`. If that function fails, sets failbit.

Definition at line 290 of file ostream.tcc.

References std::ios_base::badbit, std::basic_ios<_CharT, _Traits>::fail(), std::ios_base::failbit, std::ios_base::goodbit, std::ios_base::out, std::basic_ios<_CharT, _Traits>::rdbuf(), and std::basic_ios<_CharT, _Traits>::setstate().

5.611.5.90 **fmtflags** std::ios_base::setf (**fmtflags** __fmtfl) [inline],[inherited]

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
----------------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set.

Definition at line 646 of file ios_base.h.

Referenced by std::boolalpha(), std::dec(), std::fixed(), std::hex(), std::hexfloat(), std::left(), std::oct(), std::__detail<::operator>>(), std::right(), std::scientific(), std::showbase(), std::showpoint(), std::showpos(), std::skipws(), std::unitbuf(), and std::uppercase().

5.611.5.91 **fmtflags** std::ios_base::setf (**fmtflags** __fmtfl, **fmtflags** __mask) [inline],[inherited]

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
<code>__mask</code>	The flags mask for <i>fmtfl</i> .

Returns

The previous format control flags.

This function clears *mask* in the format flags, then sets *fmtfl* & *mask*. An example mask is `ios_base::adjustfield`.

Definition at line 663 of file ios_base.h.

5.611.5.92 **template**<typename _CharT, typename _Traits> void std::basic_ios<_CharT, _Traits>::setstate (**istate** __state) [inline],[inherited]

Sets additional flags in the error state.

Parameters

<code>__state</code>	The additional state flag(s) to set.
----------------------	--------------------------------------

See `std::ios_base::iostate` for the possible bit values.

Definition at line 157 of file `basic_ios.h`.

Referenced by `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::tr2::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::ws()`.

5.611.5.93 `template<typename _CharT, typename _Traits > int std::basic_istream< _CharT, _Traits >::sync (void)`
[inherited]

Synchronizing the stream buffer.

Returns

0 on success, -1 on failure

If `rdbuf()` is a null pointer, returns -1.

Otherwise, calls `rdbuf()->pubsync()`, and if that returns -1, sets `badbit` and returns -1.

Otherwise, returns 0.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

Definition at line 781 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::goodbit`, `std::basic_streambuf< _CharT, _Traits >::pubsync()`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_ios< _CharT, _Traits >::setstate()`, and `std::basic_istream< _CharT, _Traits >::tellg()`.

Referenced by `std::basic_istream< _CharT, _Traits >::unget()`.

5.611.5.94 `static bool std::ios_base::sync_with_stdio (bool __sync = true)` [static], [inherited]

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., stdout) and the standard C++ objects (e.g., cout). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstream.html#std.io.filestreams.binary>

5.611.5.95 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits>::pos_type
std::basic_istream<_CharT, _Traits>::tellg (void) [inherited]`

Getting the current read position.

Returns

A file position object.

If `fail()` is not false, returns `pos_type(-1)` to indicate failure. Otherwise returns `rdbuf()->pubseekoff(0, cur, in)`.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`. At variance with `putback`, `unget` and `seekg`, eofbit is not cleared first.

Definition at line 817 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::cur`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::ios_base::in`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_istream<_CharT, _Traits>::seekg()`.

Referenced by `std::basic_istream<_CharT, _Traits>::sync()`.

5.611.5.96 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>::pos_type
std::basic_ostream<_CharT, _Traits>::tellp () [inherited]`

Getting the current write position.

Returns

A file position object.

If `fail()` is not false, returns `pos_type(-1)` to indicate failure. Otherwise returns `rdbuf()->pubseekoff(0, cur, out)`.

Definition at line 237 of file `ostream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::cur`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::ios_base::out`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ostream<_CharT, _Traits>::seekp()`.

Referenced by `std::basic_ostream<_CharT, _Traits>::flush()`.

```
5.611.5.97 template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>* std::basic_ios<_CharT,
_Traits>::tie ( ) const [inline], [inherited]
```

Fetches the current *tied* stream.

Returns

A pointer to the tied stream, or NULL if the stream is not tied.

A stream may be *tied* (or synchronized) to a second output stream. When this stream performs any I/O, the tied stream is first flushed. For example, `std::cin` is tied to `std::cout`.

Definition at line 295 of file `basic_ios.h`.

Referenced by `std::basic_ios<char, _Traits>::basic_ios()`, `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`, and `std::basic_istream<_CharT, _Traits>::sentry::sentry()`.

```
5.611.5.98 template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>* std::basic_ios<_CharT,
_Traits>::tie ( basic_ostream<_CharT, _Traits>* __tiestr ) [inline], [inherited]
```

Ties this stream to an output stream.

Parameters

<code>__tiestr</code>	The output stream.
-----------------------	--------------------

Returns

The previously tied output stream, or NULL if the stream was not tied.

This sets up a new tie; see `tie()` for more.

Definition at line 307 of file `basic_ios.h`.

```
5.611.5.99 template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<
_CharT, _Traits>::unget ( void ) [inherited]
```

Unextracting the previous character.

Returns

`*this`

If `rdbuf()` is not null, calls `rdbuf()->sungetc(c)`.

If `rdbuf()` is null or if `sungetc()` fails, sets `badbit` in the error state.

Note

This function first clears eofbit. Since no characters are extracted, the next call to `gcount()` will return 0, as required by DR 60.

Definition at line 746 of file `istream.tcc`.

References `std::basic_istream< _CharT, _Traits >::_M_gcount`, `std::ios_base::badbit`, `std::basic_ios< _CharT, _Traits >::clear()`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_ios< _CharT, _Traits >::rdstate()`, `std::basic_ios< _CharT, _Traits >::setstate()`, `std::basic_streambuf< _CharT, _Traits >::sungetc()`, and `std::basic_istream< _CharT, _Traits >::sync()`.

Referenced by `std::__detail::operator>>()`, and `std::basic_istream< _CharT, _Traits >::putback()`.

5.611.5.100 `void std::ios_base::unsetf (fmtflags __mask)` `[inline]`, `[inherited]`

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

Definition at line 678 of file `ios_base.h`.

Referenced by `std::defaultfloat()`, `std::noboolalpha()`, `std::noshowbase()`, `std::noshowpoint()`, `std::noshowpos()`, `std::noskipws()`, `std::nounitbuf()`, and `std::nouppercase()`.

5.611.5.101 `template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::widen (char __c)`
`const` `[inline]`, `[inherited]`

Widens characters.

Parameters

<code>__c</code>	The character to widen.
------------------	-------------------------

Returns

The widened character.

Maps a character of `char` to a character of `char_type`.

Returns the result of

```
1 std::use_facet<ctype<char_type>> >(getloc()).widen(c)
```


Additional l10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Definition at line 449 of file `basic_ios.h`.

Referenced by `std::basic_ios< char, _Traits >::fill()`, `std::getline()`, `std::normal_distribution< _RealType >::operator()`, `std::gamma_distribution< _RealType >::operator()`, `std::negative_binomial_distribution< _IntType >::operator()`, `std::tr2::operator>>()`, and `std::operator>>()`.

5.611.5.102 `streamsize std::ios_base::width () const` `[inline]`, `[inherited]`

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Definition at line 712 of file `ios_base.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::money_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, and `std::operator>>()`.

5.611.5.103 `streamsize std::ios_base::width (streamsize __wide)` `[inline]`, `[inherited]`

Changing flags.

Parameters

<code>__wide</code>	The new width value.
---------------------	----------------------

Returns

The previous value of `width()`.

Definition at line 721 of file `ios_base.h`.

5.611.5.104 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::write (const char_type * __s, streamsize __n)` `[inherited]`

Character string insertion.

Parameters

<code>__s</code>	The array to insert.
<code>__n</code>	Maximum number of characters to insert.

Returns

*this

Characters are copied from ___s and inserted into the stream until one of the following happens:

- ___n characters are inserted
- inserting into the output sequence fails (in this case, badbit will be set in the stream's error state)

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 183 of file ostream.tcc.

References std::basic_ostream< _CharT, _Traits >::_M_write(), std::ios_base::badbit, and std::basic_ostream< _CharT, _Traits >::flush().

Referenced by std::basic_ostream< _CharT, _Traits >::put().

5.611.5.105 static int std::ios_base::xalloc () throw [static], [inherited]

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the iword and pword functions. The expectation is that an application calls xalloc in order to obtain an index in the iword and pword arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. xalloc is guaranteed to return an index that is safe to use in the iword and pword arrays.

5.611.6 Member Data Documentation

5.611.6.1 template<typename _CharT, typename _Traits> streamsize std::basic_istream< _CharT, _Traits >::_M_gcount [protected], [inherited]

The number of characters extracted in the previous unformatted function; see gcount().

Definition at line 82 of file istream.

Referenced by std::basic_istream< _CharT, _Traits >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::basic_istream< _CharT, _Traits >::ignore(), std::basic_istream< _CharT, _Traits >::peek(), std::basic_istream< _CharT, _Traits >::putback(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::readsome(), and std::basic_istream< _CharT, _Traits >::unget().

5.611.6.2 `const fmtflags std::ios_base::adjustfield` `[static], [inherited]`

A mask of left|right|internal. Useful for the 2-arg form of `setf`.

Definition at line 378 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _Inlter >::do_get()`, `std::num_put< _CharT, _Outlter >::do_put()`, `std::internal()`, `std::left()`, and `std::right()`.

5.611.6.3 `const openmode std::ios_base::app` `[static], [inherited]`

Seek to end before each write.

Definition at line 432 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

5.611.6.4 `const openmode std::ios_base::ate` `[static], [inherited]`

Open and seek to end immediately after opening.

Definition at line 435 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::open()`.

5.611.6.5 `const iostate std::ios_base::badbit` `[static], [inherited]`

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Definition at line 402 of file `ios_base.h`.

Referenced by `std::basic_ios< char, _Traits >::bad()`, `std::basic_ios< char, _Traits >::fail()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsomewhat()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::basic_ostream< _CharT, _Traits >::write()`.

5.611.6.6 `const fmtflags std::ios_base::basefield` `[static], [inherited]`

A mask of dec|oct|hex. Useful for the 2-arg form of `setf`.

Definition at line 381 of file `ios_base.h`.

Referenced by `std::dec()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::num_put< _CharT, _Outlter >::do_put()`, `std::hex()`, `std::oct()`, and `std::basic_ostream< _CharT, _Traits >::operator<<()`.

5.611.6.7 const seekdir std::ios_base::beg [static],[inherited]

Request a seek relative to the beginning of the stream.

Definition at line 464 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::seekpos(), and __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync().

5.611.6.8 const openmode std::ios_base::binary [static],[inherited]

Perform input and output in binary mode (as opposed to text mode). This is probably not what you think it is; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>.

Definition at line 440 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::showmanyc().

5.611.6.9 const fmtflags std::ios_base::boolalpha [static],[inherited]

Insert/extract `bool` in alphabetic rather than numeric format.

Definition at line 326 of file ios_base.h.

Referenced by std::boolalpha(), std::num_get< _CharT, _InIter >::do_get(), std::num_put< _CharT, _OutIter >::do_put(), and std::noboolalpha().

5.611.6.10 const seekdir std::ios_base::cur [static],[inherited]

Request a seek relative to the current position within the sequence.

Definition at line 467 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::imbue(), std::basic_filebuf< _CharT, _Traits >::overflow(), std::basic_filebuf< _CharT, _Traits >::pbackfail(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff(), std::basic_filebuf< _CharT, _Traits >::seekoff(), __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync(), std::basic_istream< _CharT, _Traits >::tellg(), and std::basic_ostream< _CharT, _Traits >::tellp().

5.611.6.11 const fmtflags std::ios_base::dec [static],[inherited]

Converts integer input or generates integer output in decimal base.

Definition at line 329 of file ios_base.h.

Referenced by std::dec().

5.611.6.12 `const seekdir std::ios_base::end` `[static], [inherited]`

Request a seek relative to the current end of the sequence.

Definition at line 470 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::open()`, and `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`.

5.611.6.13 `const iostate std::ios_base::eofbit` `[static], [inherited]`

Indicates that an input operation reached the end of an input sequence.

Definition at line 405 of file `ios_base.h`.

Referenced by `std::time_get< _CharT, _Inlter >::do_get()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::time_get< _CharT, _Inlter >::do_get_date()`, `std::time_get< _CharT, _Inlter >::do_get_monthname()`, `std::time_get< _CharT, _Inlter >::do_get_time()`, `std::time_get< _CharT, _Inlter >::do_get_weekday()`, `std::time_get< _CharT, _Inlter >::do_get_year()`, `std::basic_ios< char, _Traits >::eof()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::time_get< _CharT, _Inlter >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::ws()`.

5.611.6.14 `const iostate std::ios_base::failbit` `[static], [inherited]`

Indicates that an input operation failed to read the expected characters, or that an output operation failed to generate the desired characters.

Definition at line 410 of file `ios_base.h`.

Referenced by `std::time_get< _CharT, _Inlter >::do_date_order()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::time_get< _CharT, _Inlter >::do_get_monthname()`, `std::time_get< _CharT, _Inlter >::do_get_weekday()`, `std::time_get< _CharT, _Inlter >::do_get_year()`, `std::basic_ios< char, _Traits >::fail()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::time_get< _CharT, _Inlter >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_istream< _CharT, _Traits >::sentry::sentry()`.

5.611.6.15 `const fmtflags std::ios_base::fixed` `[static], [inherited]`

Generate floating-point output in fixed-point notation.

Definition at line 332 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _Inlter >::do_get()`, `std::fixed()`, and `std::hexfloat()`.

5.611.6.16 `const fmtflags std::ios_base::floatfield` `[static], [inherited]`

A mask of scientific|fixed. Useful for the 2-arg form of `setf`.

Definition at line 384 of file `ios_base.h`.

Referenced by `std::defaultfloat()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::fixed()`, `std::hexfloat()`, and `std::scientific()`.

5.611.6.17 `const iostate std::ios_base::goodbit` `[static], [inherited]`

Indicates all is well.

Definition at line 413 of file `ios_base.h`.

Referenced by `std::time_get< _CharT, _Inlter >::do_date_order()`, `std::time_get< _CharT, _Inlter >::do_get()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::time_get< _CharT, _Inlter >::do_get_monthname()`, `std::time_get< _CharT, _Inlter >::do_get_weekday()`, `std::time_get< _CharT, _Inlter >::do_get_year()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::time_get< _CharT, _Inlter >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_ios< char, _Traits >::rdstate()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sync()`, and `std::basic_istream< _CharT, _Traits >::unset()`.

5.611.6.18 `const fmtflags std::ios_base::hex` `[static], [inherited]`

Converts integer input or generates integer output in hexadecimal base.

Definition at line 335 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _Inlter >::do_get()`, `std::num_put< _CharT, _Outlter >::do_put()`, `std::hex()`, and `std::basic_ostream< _CharT, _Traits >::operator<<()`.

5.611.6.19 `const openmode std::ios_base::in` `[static], [inherited]`

Open for input. Default for `ifstream` and `fstream`.

Definition at line 443 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos()`, `std::basic_filebuf< _CharT, _Traits >::showmanyc()`, `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::underflow()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

5.611.6.20 `const fmtflags std::ios_base::internal` `[static], [inherited]`

Adds fill characters at a designated internal point in certain generated output, or identical to `right` if no such point is designated.

Definition at line 340 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, and `std::ios_base::internal()`.

5.611.6.21 `const fmtflags std::ios_base::left` `[static], [inherited]`

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.)

Definition at line 344 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, and `std::ios_base::left()`.

5.611.6.22 `const fmtflags std::ios_base::oct` `[static], [inherited]`

Converts integer input or generates integer output in octal base.

Definition at line 347 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _InIter >::do_get()`, `std::oct()`, and `std::basic_ostream< _CharT, _Traits >::operator<<()`.

5.611.6.23 `const openmode std::ios_base::out` `[static], [inherited]`

Open for output. Default for `ofstream` and `fstream`.

Definition at line 446 of file `ios_base.h`.

Referenced by `std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow()`, `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::pbackfail()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos()`, `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, and `std::basic_filebuf< _CharT, _Traits >::xspn()`.

5.611.6.24 `const fmtflags std::ios_base::right` `[static], [inherited]`

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Definition at line 351 of file `ios_base.h`.

Referenced by `std::right()`.

5.611.6.25 `const fmtflags std::ios_base::scientific` `[static], [inherited]`

Generates floating-point output in scientific notation.

Definition at line 354 of file `ios_base.h`.

Referenced by `std::hexfloat()`, and `std::scientific()`.

5.611.6.26 `const fmtflags std::ios_base::showbase` `[static], [inherited]`

Generates a prefix indicating the numeric base of generated integer output.

Definition at line 358 of file `ios_base.h`.

Referenced by `std::money_get<_CharT, _InIter>::do_get()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_put<_CharT, _OutIter>::do_put()`, `std::noshowbase()`, and `std::showbase()`.

5.611.6.27 `const fmtflags std::ios_base::showpoint` `[static], [inherited]`

Generates a decimal-point character unconditionally in generated floating-point output.

Definition at line 362 of file `ios_base.h`.

Referenced by `std::noshowpoint()`, and `std::showpoint()`.

5.611.6.28 `const fmtflags std::ios_base::showpos` `[static], [inherited]`

Generates a + sign in non-negative generated numeric output.

Definition at line 365 of file `ios_base.h`.

Referenced by `std::num_get<_CharT, _InIter>::do_get()`, `std::noshowpos()`, and `std::showpos()`.

5.611.6.29 `const fmtflags std::ios_base::skipws` `[static], [inherited]`

Skips leading white space before certain input operations.

Definition at line 368 of file `ios_base.h`.

Referenced by `std::noskipws()`, `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, and `std::skipws()`.

5.611.6.30 `const openmode std::ios_base::trunc` `[static], [inherited]`

Open for input. Default for `ofstream`.

Definition at line 449 of file `ios_base.h`.

5.611.6.31 `const fmtflags std::ios_base::unitbuf` `[static], [inherited]`

Flushes output after each output operation.

Definition at line 371 of file `ios_base.h`.

Referenced by `std::nunitbuf()`, and `std::unitbuf()`.

5.611.6.32 `const fmtflags std::ios_base::uppercase` `[static], [inherited]`

Replaces certain lowercase letters with their uppercase equivalents in generated output.

Definition at line 375 of file `ios_base.h`.

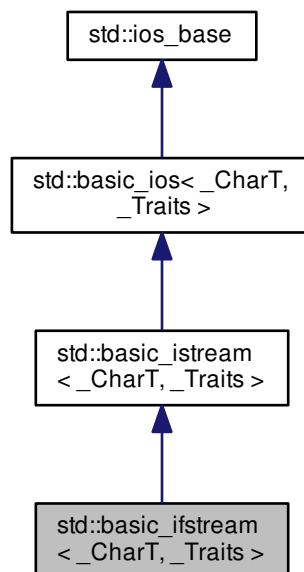
Referenced by `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::nouppercase()`, and `std::uppercase()`.

The documentation for this class was generated from the following file:

- [fstream](#)

5.612 `std::basic_ifstream< _CharT, _Traits >` Class Template Reference

Inheritance diagram for `std::basic_ifstream< _CharT, _Traits >`:



Public Types

- typedef [ctype](#)< _CharT > **__ctype_type**
 - typedef [basic_filebuf](#)< char_type, traits_type > **__filebuf_type**
 - typedef [basic_ios](#)< _CharT, _Traits > **__ios_type**
 - typedef [basic_istream](#)< char_type, traits_type > **__istream_type**
 - typedef [num_get](#)< _CharT, [istreambuf_iterator](#)< _CharT, _Traits > > **__num_get_type**
 - typedef [basic_streambuf](#)< _CharT, _Traits > **__streambuf_type**
 - typedef _CharT **char_type**
 - enum [event](#) { [erase_event](#), [imbue_event](#), [copyfmt_event](#) }
 - typedef void(* [event_callback](#)) (event __e, [ios_base](#) & __b, int __i)
 - typedef _ios_Fmtflags **fmtflags**
 - typedef traits_type::int_type **int_type**
 - typedef int **io_state**
 - typedef _ios_istate **iostate**
 - typedef traits_type::off_type **off_type**
 - typedef int **open_mode**
 - typedef _ios_Openmode **openmode**
 - typedef traits_type::pos_type **pos_type**
 - typedef int **seek_dir**
 - typedef _ios_Seekdir **seekdir**
 - typedef [std::streamoff](#) **streamoff**
 - typedef [std::streampos](#) **streampos**
 - typedef _Traits **traits_type**
-
- typedef [num_put](#)< _CharT, [ostreambuf_iterator](#)< _CharT, _Traits > > **__num_put_type**

Public Member Functions

- [basic_ifstream](#) ()
- [basic_ifstream](#) (const char * __s, [ios_base::openmode](#) __mode=[ios_base::in](#))
- [basic_ifstream](#) (const [std::string](#) & __s, [ios_base::openmode](#) __mode=[ios_base::in](#))
- **basic_ifstream** (const [basic_ifstream](#) &)=delete
- **basic_ifstream** ([basic_ifstream](#) && __rhs)
- [~basic_ifstream](#) ()
- template<typename _ValueT >
 [basic_istream](#)< _CharT, _Traits > & **_M_extract** (_ValueT & __v)
- const [locale](#) & **_M_getloc** () const
- void **_M_setstate** ([iostate](#) __state)
- bool **bad** () const
- void **clear** ([iostate](#) __state=[goodbit](#))
- void **close** ()
- [basic_ios](#) & **copyfmt** (const [basic_ios](#) & __rhs)
- bool **eof** () const
- [iostate](#) **exceptions** () const
- void **exceptions** ([iostate](#) __except)
- bool **fail** () const
- char_type **fill** () const
- char_type **fill** (char_type __ch)

- `fmtflags flags () const`
 - `fmtflags flags (fmtflags __fmtfl)`
 - `streamsize gcount () const`
 - `template<>`
`basic_istream< char > & getline (char_type *__s, streamsize __n, char_type __delim)`
 - `template<>`
`basic_istream< wchar_t > & getline (char_type *__s, streamsize __n, char_type __delim)`
 - `locale getloc () const`
 - `bool good () const`
 - `template<>`
`basic_istream< char > & ignore (streamsize __n)`
 - `template<>`
`basic_istream< char > & ignore (streamsize __n, int_type __delim)`
 - `template<>`
`basic_istream< wchar_t > & ignore (streamsize __n)`
 - `template<>`
`basic_istream< wchar_t > & ignore (streamsize __n, int_type __delim)`
 - `locale imbue (const locale &__loc)`
 - `bool is_open ()`
 - `bool is_open () const`
 - `long & iword (int __ix)`
 - `char narrow (char_type __c, char __default) const`
 - `void open (const char *__s, ios_base::openmode __mode=ios_base::in)`
 - `void open (const std::string &__s, ios_base::openmode __mode=ios_base::in)`
 - `basic_ifstream & operator= (const basic_ifstream &)=delete`
 - `basic_ifstream & operator= (basic_ifstream && __rhs)`
 - `__istream_type & operator>> (void *&__p)`
 - `__istream_type & operator>> (__streambuf_type *__sb)`
 - `streamsize precision () const`
 - `streamsize precision (streamsize __prec)`
 - `void *& pword (int __ix)`
 - `basic_streambuf< _CharT, _Traits > * rdbuf (basic_streambuf< _CharT, _Traits > *__sb)`
 - `__filebuf_type * rdbuf () const`
 - `iosate rdstate () const`
 - `void register_callback (event_callback __fn, int __index)`
 - `fmtflags setf (fmtflags __fmtfl)`
 - `fmtflags setf (fmtflags __fmtfl, fmtflags __mask)`
 - `void setstate (iosate __state)`
 - `void swap (basic_ifstream &__rhs)`
 - `basic_ostream< _CharT, _Traits > * tie () const`
 - `basic_ostream< _CharT, _Traits > * tie (basic_ostream< _CharT, _Traits > *__tiestr)`
 - `void unsetf (fmtflags __mask)`
 - `char_type widen (char __c) const`
 - `streamsize width () const`
 - `streamsize width (streamsize __wide)`
-
- `__istream_type & operator>> (__istream_type &(*__pf)(__istream_type &))`
 - `__istream_type & operator>> (__ios_type &(*__pf)(__ios_type &))`
 - `__istream_type & operator>> (ios_base &(*__pf)(ios_base &))`

Extractors

All the `operator>>` functions (aka formatted input functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_ifstream::sentry` with the second argument (`noskipws`) set to `false`. This has several effects, concluding with the setting of a status flag; see the `sentry` documentation for more.

If the `sentry` status is good, the function tries to extract whatever data is appropriate for the type of the argument.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- `__istream_type & operator>> (bool &__n)`
- `__istream_type & operator>> (short &__n)`
- `__istream_type & operator>> (unsigned short &__n)`
- `__istream_type & operator>> (int &__n)`
- `__istream_type & operator>> (unsigned int &__n)`
- `__istream_type & operator>> (long &__n)`
- `__istream_type & operator>> (unsigned long &__n)`
- `__istream_type & operator>> (long long &__n)`
- `__istream_type & operator>> (unsigned long long &__n)`

- `__istream_type & operator>> (float &__f)`
- `__istream_type & operator>> (double &__f)`
- `__istream_type & operator>> (long double &__f)`

Unformatted Input Functions

All the unformatted input functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_ifstream::sentry` with the second argument (`noskipws`) set to `true`. This has several effects, concluding with the setting of a status flag; see the `sentry` documentation for more.

If the `sentry` status is good, the function tries to extract whatever data is appropriate for the type of the argument.

The number of characters extracted is stored for later retrieval by `gcount()`.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- `int_type get ()`
- `__istream_type & get (char_type &__c)`
- `__istream_type & get (char_type *__s, streamsize __n, char_type __delim)`
- `__istream_type & get (char_type *__s, streamsize __n)`
- `__istream_type & get (__streambuf_type &__sb, char_type __delim)`
- `__istream_type & get (__streambuf_type &__sb)`
- `__istream_type & getline (char_type *__s, streamsize __n, char_type __delim)`
- `__istream_type & getline (char_type *__s, streamsize __n)`
- `__istream_type & ignore (streamsize __n, int_type __delim)`
- `__istream_type & ignore (streamsize __n)`
- `__istream_type & ignore ()`
- `int_type peek ()`
- `__istream_type & read (char_type *__s, streamsize __n)`
- `streamsize readsome (char_type *__s, streamsize __n)`
- `__istream_type & putback (char_type __c)`
- `__istream_type & unget ()`
- `int sync ()`
- `pos_type tellg ()`
- `__istream_type & seekg (pos_type)`
- `__istream_type & seekg (off_type, ios_base::seekdir)`

- `operator bool () const`
- `bool operator! () const`

Static Public Member Functions

- static bool [sync_with_stdio](#) (bool __sync=true)
- static int [xalloc](#) () throw ()

Static Public Attributes

- static const [fmtflags](#) [adjustfield](#)
- static const [openmode](#) [app](#)
- static const [openmode](#) [ate](#)
- static const [iostate](#) [badbit](#)
- static const [fmtflags](#) [basefield](#)
- static const [seekdir](#) [beg](#)
- static const [openmode](#) [binary](#)
- static const [fmtflags](#) [boolalpha](#)
- static const [seekdir](#) [cur](#)
- static const [fmtflags](#) [dec](#)
- static const [seekdir](#) [end](#)
- static const [iostate](#) [eofbit](#)
- static const [iostate](#) [failbit](#)
- static const [fmtflags](#) [fixed](#)
- static const [fmtflags](#) [floatfield](#)
- static const [iostate](#) [goodbit](#)
- static const [fmtflags](#) [hex](#)
- static const [openmode](#) [in](#)
- static const [fmtflags](#) [internal](#)
- static const [fmtflags](#) [left](#)
- static const [fmtflags](#) [oct](#)
- static const [openmode](#) [out](#)
- static const [fmtflags](#) [right](#)
- static const [fmtflags](#) [scientific](#)
- static const [fmtflags](#) [showbase](#)
- static const [fmtflags](#) [showpoint](#)
- static const [fmtflags](#) [showpos](#)
- static const [fmtflags](#) [skipws](#)
- static const [openmode](#) [trunc](#)
- static const [fmtflags](#) [unitbuf](#)
- static const [fmtflags](#) [uppercase](#)

Protected Types

- enum { **[_S_local_word_size](#)** }

Protected Member Functions

- void **_M_cache_locale** (const [locale](#) &__loc)
- void **_M_call_callbacks** ([event](#) __ev) throw ()
- void **_M_dispose_callbacks** (void) throw ()
- template<typename _ValueT >
[__istream_type](#) & **_M_extract** (_ValueT &__v)
- [_Words](#) & **_M_grow_words** (int __index, bool __iword)
- void **_M_init** () throw ()
- void **_M_move** ([ios_base](#) &) noexcept
- void **_M_swap** ([ios_base](#) &__rhs) noexcept
- void **init** ([basic_streambuf](#)< _CharT, _Traits > *__sb)
- void **move** ([basic_ios](#) &__rhs)
- void **move** ([basic_ios](#) &&__rhs)
- void **set_rdbuf** ([basic_streambuf](#)< _CharT, _Traits > *__sb)
- void **swap** ([basic_ios](#) &__rhs) noexcept
- void **swap** ([basic_istream](#) &__rhs)

Protected Attributes

- [_Callback_list](#) * **_M_callbacks**
- const [__ctype_type](#) * **_M_ctype**
- [iostate](#) **_M_exception**
- [char_type](#) **_M_fill**
- bool **_M_fill_init**
- [fmtflags](#) **_M_flags**
- [streamsize](#) **_M_gcount**
- [locale](#) **_M_ios_locale**
- [_Words](#) **_M_local_word** [[_S_local_word_size](#)]
- const [__num_get_type](#) * **_M_num_get**
- const [__num_put_type](#) * **_M_num_put**
- [streamsize](#) **_M_precision**
- [basic_streambuf](#)< _CharT, _Traits > * **_M_streambuf**
- [iostate](#) **_M_streambuf_state**
- [basic_ostream](#)< _CharT, _Traits > * **_M_tie**
- [streamsize](#) **_M_width**
- [_Words](#) * **_M_word**
- int **_M_word_size**
- [_Words](#) **_M_word_zero**

5.612.1 Detailed Description

```
template<typename _CharT, typename _Traits>
class std::basic_ifstream< _CharT, _Traits >
```

Controlling input for files.

Template Parameters

<code>_CharT</code>	Type of character stream.
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .

This class supports reading from named files, using the inherited functions from `std::basic_istream`. To control the associated sequence, an instance of `std::basic_filebuf` is used, which this page refers to as `sb`.

Definition at line 455 of file `fstream`.

5.612.2 Member Typedef Documentation

5.612.2.1 `template<typename _CharT, typename _Traits> typedef num_put<_CharT, ostreambuf_iterator<_CharT, _Traits>> std::basic_ios<_CharT, _Traits>::__num_put_type [inherited]`

These are non-standard types.

Definition at line 89 of file `basic_ios.h`.

5.612.2.2 `typedef void(* std::ios_base::event_callback)(event __e, ios_base &__b, int __i) [inherited]`

The type of an event callback function.

Parameters

<code>__e</code>	One of the members of the event enum.
<code>__b</code>	Reference to the <code>ios_base</code> object.
<code>__i</code>	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several `ios_base` and `basic_ios` functions, specifically `imbue()`, `copyfmt()`, and `~ios()`.

Definition at line 504 of file `ios_base.h`.

5.612.2.3 `typedef _Ios_Fmtflags std::ios_base::fmtflags [inherited]`

This is a bitmask type.

`_Ios_Fmtflags` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `fmtflags` are:

- `boolalpha`

- `dec`
- `fixed`
- `hex`
- `internal`
- `left`
- `oct`
- `right`
- `scientific`
- `showbase`
- `showpoint`
- `showpos`
- `skipws`
- `unitbuf`
- `uppercase`
- `adjustfield`
- `basefield`
- `floatfield`

Definition at line 323 of file `ios_base.h`.

5.612.2.4 `typedef _ios_iostate std::ios_base::iostate` `[inherited]`

This is a bitmask type.

`_Ios_Iostate` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `iostate` are:

- `badbit`
- `eofbit`
- `failbit`
- `goodbit`

Definition at line 398 of file `ios_base.h`.

5.612.2.5 `typedef _Ios_Openmode std::ios_base::openmode` [inherited]

This is a bitmask type.

`_Ios_Openmode` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `openmode` are:

- `app`
- `ate`
- `binary`
- `in`
- `out`
- `trunc`

Definition at line 429 of file `ios_base.h`.

5.612.2.6 `typedef _Ios_Seekdir std::ios_base::seekdir` [inherited]

This is an enumerated type.

`_Ios_Seekdir` is implementation-defined. Defined values of type `seekdir` are:

- `beg`
- `cur`, equivalent to `SEEK_CUR` in the C standard library.
- `end`, equivalent to `SEEK_END` in the C standard library.

Definition at line 461 of file `ios_base.h`.

5.612.3 Member Enumeration Documentation

5.612.3.1 `enum std::ios_base::event` [inherited]

The set of events that may be passed to an event callback.

`erase_event` is used during `~ios()` and `copyfmt()`. `imbue_event` is used during `imbue()`. `copyfmt_event` is used during `copyfmt()`.

Definition at line 487 of file `ios_base.h`.

5.612.4 Constructor & Destructor Documentation

5.612.4.1 `template<typename _CharT, typename _Traits> std::basic_ifstream<_CharT, _Traits>::basic_ifstream ()` [inline]

Default constructor.

Initializes `sb` using its default constructor, and passes `&sb` to the base class initializer. Does not open any files (you haven't given it a filename to open).

Definition at line 481 of file `fstream`.

5.612.4.2 `template<typename _CharT, typename _Traits> std::basic_ifstream<_CharT, _Traits>::basic_ifstream (const char * __s, ios_base::openmode __mode = ios_base::in)` [inline], [explicit]

Create an input file stream.

Parameters

<code>__s</code>	Null terminated string specifying the filename.
<code>__mode</code>	Open file in specified mode (see std::ios_base).

`ios_base::in` is automatically included in `__mode`.

Tip: When using `std::string` to hold the filename, you must use `.c_str()` before passing it to this constructor.

Definition at line 495 of file `fstream`.

5.612.4.3 `template<typename _CharT, typename _Traits> std::basic_ifstream<_CharT, _Traits>::basic_ifstream (const std::string & __s, ios_base::openmode __mode = ios_base::in) [inline], [explicit]`

Create an input file stream.

Parameters

<code>__s</code>	<code>std::string</code> specifying the filename.
<code>__mode</code>	Open file in specified mode (see std::ios_base).

`ios_base::in` is automatically included in `__mode`.

Definition at line 511 of file `fstream`.

5.612.4.4 `template<typename _CharT, typename _Traits> std::basic_ifstream<_CharT, _Traits>::~basic_ifstream () [inline]`

The destructor does nothing.

The file is closed by the filebuf object, not the formatting stream.

Definition at line 533 of file `fstream`.

5.612.5 Member Function Documentation

5.612.5.1 `const locale& std::ios_base::M_getloc () const [inline], [inherited]`

Locale access.

Returns

A reference to the current locale.

Like `getloc` above, but returns a reference instead of generating a copy.

Definition at line 774 of file `ios_base.h`.

Referenced by `std::time_get<_CharT, _Inlter>::do_date_order()`, `std::time_get<_CharT, _Inlter>::do_get()`, `std::money_get<_CharT, _Inlter>::do_get()`, `std::num_get<_CharT, _Inlter>::do_get()`, `std::time_get<_CharT, _Inlter>::do_get_date()`, `std::time_get<_CharT, _Inlter>::do_get_monthname()`, `std::time_get<_CharT, _Inlter>::do_get_time()`, `std::time_get<_CharT, _Inlter>::do_get_weekday()`, `std::time_get<_CharT, _Inlter>::do_get_year()`, `std::time_put<_CharT, _Outlter>::do_put()`, `std::num_put<_CharT, _Outlter>::do_put()`, `std::time_get<_CharT, _Inlter>::get()`, and `std::time_put<_CharT, _Outlter>::put()`.

5.612.5.2 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::bad () const`
`[inline], [inherited]`

Fast error checking.

Returns

True if the badbit is set.

Note that other iostate flags may also be set.

Definition at line 211 of file `basic_ios.h`.

5.612.5.3 `template<typename _CharT, typename _Traits > void std::basic_ios< _CharT, _Traits >::clear (iostate __state =`
`goodbit) [inherited]`

[Re]sets the error state.

Parameters

<code>__state</code>	The new state flag(s) to set.
----------------------	-------------------------------

See `std::ios_base::iostate` for the possible bit values. Most users will not need to pass an argument.

Definition at line 41 of file `basic_ios.tcc`.

Referenced by `std::basic_ios< char, _Traits >::exceptions()`, `std::__detail::operator>>()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_ios< char, _Traits >::rdstate()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ios< char, _Traits >::setstate()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

5.612.5.4 `template<typename _CharT, typename _Traits > void std::basic_ifstream< _CharT, _Traits >::close ()`
`[inline]`

Close the file.

Calls `std::basic_filebuf::close()`. If that function fails, `failbit` is set in the stream's error state.

Definition at line 633 of file `fstream`.

5.612.5.5 `template<typename _CharT, typename _Traits > basic_ios< _CharT, _Traits > & std::basic_ios< _CharT, _Traits`
`>::copyfmt (const basic_ios< _CharT, _Traits > & __rhs) [inherited]`

Copies fields of `__rhs` into this.

Parameters

<code>__rhs</code>	The source values for the copies.
--------------------	-----------------------------------

Returns

Reference to this object.

All fields of `__rhs` are copied into this object except that `rdbuf()` and `rdstate()` remain unchanged. All values in the `pword` and `iword` arrays are copied. Before copying, each callback is invoked with `erase_event`. After copying, each (new) callback is invoked with `copyfmt_event`. The final step is to copy exceptions().

Definition at line 63 of file `basic_ios.tcc`.

References `std::basic_ios< _CharT, _Traits >::exceptions()`, `std::basic_ios< _CharT, _Traits >::fill()`, `std::ios_base::flags()`, `std::ios_base::getloc()`, `std::ios_base::precision()`, `std::basic_ios< _CharT, _Traits >::tie()`, `std::tie()`, and `std::ios_base::width()`.

Referenced by `std::basic_ios< char, _Traits >::rdbuf()`.

```
5.612.5.6 template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::eof ( ) const
    [inline], [inherited]
```

Fast error checking.

Returns

True if the eofbit is set.

Note that other `iosstate` flags may also be set.

Definition at line 190 of file `basic_ios.h`.

```
5.612.5.7 template<typename _CharT, typename _Traits> iosstate std::basic_ios< _CharT, _Traits >::exceptions ( ) const
    [inline], [inherited]
```

Throwing exceptions on errors.

Returns

The current exceptions mask.

This changes nothing in the stream. See the one-argument version of `exceptions(iosstate)` for the meaning of the return value.

Definition at line 222 of file `basic_ios.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, and `std::basic_ios< char, _Traits >::setstate()`.

```
5.612.5.8 template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits >::exceptions ( iosstate
    __except ) [inline], [inherited]
```

Throwing exceptions on errors.

Parameters

<code>__except</code>	The new exceptions mask.
-----------------------	--------------------------

By default, error flags are set silently. You can set an exceptions mask for each stream; if a bit in the mask becomes set in the error flags, then an exception of type `std::ios_base::failure` is thrown.

If the error flag is already set when the exceptions mask is added, the exception is immediately thrown. Try running the following under GCC 3.1 or later:

```

1 #include <iostream>
2 #include <fstream>
3 #include <exception>
4
5 int main()
6 {
7     std::set_terminate (__gnu_cxx::__verbose_terminate_handler);
8
9     std::ifstream f ("/etc/motd");
10
11     std::cerr << "Setting badbit\n";
12     f.setstate (std::ios_base::badbit);
13
14     std::cerr << "Setting exception mask\n";
15     f.exceptions (std::ios_base::badbit);
16 }
```

Definition at line 257 of file `basic_ios.h`.

5.612.5.9 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::fail () const`
`[inline], [inherited]`

Fast error checking.

Returns

True if either the badbit or the failbit is set.

Checking the badbit in `fail()` is historical practice. Note that other iostate flags may also be set.

Definition at line 201 of file `basic_ios.h`.

Referenced by `std::basic_ios< char, _Traits >::operator bool()`, `std::basic_ios< char, _Traits >::operator!()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, and `std::regex_traits< _Ch_type >::value()`.

5.612.5.10 `template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::fill () const`
`[inline], [inherited]`

Retrieves the *empty* character.

Returns

The current fill character.

It defaults to a space (' ') in the current locale.

Definition at line 370 of file `basic_ios.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::basic_ios< char, _Traits >::fill()`, `std::normal_distribution< _RealType >::operator()()`, `std::gamma_distribution< _RealType >::operator()()`, `std::negative_binomial_distribution< _IntType >::operator()()`, and `std::operator>>()`.

5.612.5.11 `template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::fill (char_type __ch) [inline], [inherited]`

Sets a new *empty* character.

Parameters

<code>__ch</code>	The new character.
-------------------	--------------------

Returns

The previous fill character.

The fill character is used to fill out space when P+ characters have been requested (e.g., via `setw`), Q characters are actually used, and Q<P. It defaults to a space (' ') in the current locale.

Definition at line 390 of file `basic_ios.h`.

5.612.5.12 `fmtflags std::ios_base::flags () const [inline], [inherited]`

Access to format flags.

Returns

The format control flags for both input and output.

Definition at line 619 of file `ios_base.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >::discard()`, `std::money_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::operator()()`, `std::discard_block_engine< _RandomNumberEngine, __p, __r >::operator()()`, `std::shuffle_order_engine< _RandomNumberEngine, __k >::operator()()`, `std::normal_distribution< _RealType >::operator()()`, `std::gamma_distribution< _RealType >::operator()()`, `std::binomial_distribution< _IntType >::operator()()`, `std::negative_binomial_distribution< _IntType >::operator()()`, `std::poisson_distribution< _IntType >::operator()()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, `std::__detail::operator>>()`, `std::operator>>()`, `std::linear_congruential_engine< _UIntType, __a, __c, __m >::seed()`, and `std::basic_istream< _CharT, _Traits >::sentry::sentry()`.

5.612.5.13 `fmtflags std::ios_base::flags (fmtflags __fmtfl) [inline], [inherited]`

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

Definition at line 630 of file `ios_base.h`.

5.612.5.14 `template<typename _CharT, typename _Traits> streamsize std::basic_istream< _CharT, _Traits >::gcount ()`
`const [inline],[inherited]`

Character counting.

Returns

The number of characters extracted by the previous unformatted input function dispatched for this stream.

Definition at line 269 of file `istream`.

5.612.5.15 `template<typename _CharT , typename _Traits > basic_istream< _CharT, _Traits >::int_type`
`std::basic_istream< _CharT, _Traits >::get (void) [inherited]`

Simple extraction.

Returns

A character, or `eof()`.

Tries to extract a character. If none are available, sets failbit and returns `traits::eof()`.

Definition at line 236 of file `istream.tcc`.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

Referenced by `std::basic_istream< _CharT, _Traits >::get()`, and `std::basic_istream< _CharT, _Traits >::operator>>()`.

5.612.5.16 `template<typename _CharT, typename _Traits > basic_istream< _CharT, _Traits > & std::basic_istream<`
`_CharT, _Traits >::get (char_type & __c) [inherited]`

Simple extraction.

Parameters

<code>__c</code>	The character in which to store data.
------------------	---------------------------------------

Returns

*this

Tries to extract a character and store it in `__c`. If none are available, sets failbit and returns `traits::eof()`.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 272 of file `istream.tcc`.

References `std::basic_istream< _CharT, _Traits >::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::basic_istream< _CharT, _Traits >::get()`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

5.612.5.17 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::get (char_type * __s, streamsize __n, char_type __delim)` [inherited]

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
<code>__n</code>	Maximum number of characters to store in <code>__s</code> .
<code>__delim</code>	A "stop" character.

Returns

*this

Characters are extracted and stored into `__s` until one of the following happens:

- `__n-1` characters are stored
- the input sequence reaches EOF
- the next character equals `__delim`, in which case the character is not extracted

If no characters are stored, failbit is set in the stream's error state.

In any case, a null character is stored into the next location in the array.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 309 of file `istream.tcc`.

References `std::basic_istream< _CharT, _Traits >::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::basic_istream< _CharT, _Traits >::get()`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_ios< _CharT, _Traits >::setstate()`, `std::basic_streambuf< _CharT, _Traits >::sgetc()`, and `std::basic_streambuf< _CharT, _Traits >::snextc()`.

5.612.5.18 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits >::get (char_type * __s, streamsize __n) [inline], [inherited]`

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
<code>__n</code>	Maximum number of characters to store in <i>s</i> .

Returns

*this

Returns `get(__s, __n, widen("\n"))`.

Definition at line 354 of file `istream`.

5.612.5.19 `template<typename _CharT, typename _Traits > basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::get (__streambuf_type & __sb, char_type __delim) [inherited]`

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
<code>__delim</code>	A "stop" character.

Returns

*this

Characters are extracted and inserted into `__sb` until one of the following happens:

- the input sequence reaches EOF
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted)
- the next character equals `__delim` (in this case, the character is not extracted)
- an exception occurs (and in this case is caught)

If no characters are stored, failbit is set in the stream's error state.

Definition at line 356 of file `istream.tcc`.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_ios< _CharT, _Traits >::setstate()`, `std::basic_streambuf< _CharT, _Traits >::sgetc()`, `std::basic_streambuf< _CharT, _Traits >::snextc()`, and `std::basic_streambuf< _CharT, _Traits >::sputc()`.

5.612.5.20 `template<typename _CharT, typename _Traits> __istream_type& std::basic_ifstream<_CharT, _Traits>::get (__streambuf_type &__sb) [inline], [inherited]`

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
-------------------	-------------------------------------

Returns

*this

Returns `get(__sb, widen("\n"))`.

Definition at line 387 of file istream.

5.612.5.21 `template<typename _CharT, typename _Traits> basic_ifstream<_CharT, _Traits> & std::basic_ifstream<_CharT, _Traits>::getline (char_type * __s, streamsize __n, char_type __delim) [inherited]`

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.
<code>__delim</code>	A "stop" character.

Returns

*this

Extracts and stores characters into `__s` until one of the following happens. Note that these criteria are required to be tested in the order listed here, to allow an input line to exactly fill the `__s` array without setting failbit.

1. the input sequence reaches end-of-file, in which case eofbit is set in the stream error state
2. the next character equals `__delim`, in which case the character is extracted (and therefore counted in `gcount()`) but not stored
3. `__n-1` characters are stored, in which case failbit is set in the stream error state

If no characters are extracted, failbit is set. (An empty line of input should therefore not cause failbit to be set.)

In any case, a null character is stored in the next location in the array.

Definition at line 400 of file istream.tcc.

References `std::basic_ifstream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ifstream<_CharT, _Traits>::ignore()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_streambuf<_CharT, _Traits>::sbumpc()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sgetc()`, and `std::basic_streambuf<_CharT, _Traits>::snextc()`.

Referenced by `std::basic_ifstream<_CharT, _Traits>::get()`.

5.612.5.22 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits >::getline (char_type * __s, streamsize __n) [inline],[inherited]`

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.

Returns

`*this`

Returns `getline(__s,__n,widen("\n"))`.

Definition at line 427 of file `istream`.

5.612.5.23 `template<> basic_istream< char > & std::basic_istream< char >::getline (char_type * __s, streamsize __n, char_type __delim) [inherited]`

Explicit specialization declarations, defined in `src/istream.cc`.

5.612.5.24 `locale std::ios_base::getloc () const [inline],[inherited]`

Locale access.

Returns

A copy of the current locale.

If `imbue(loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale()`, the global C++ locale.

Definition at line 763 of file `ios_base.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::money_put< _CharT, _Outiter >::do_put()`, `std::operator>>()`, and `std::ws()`.

5.612.5.25 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::good () const [inline],[inherited]`

Fast error checking.

Returns

True if no error flags are set.

A wrapper around `rdstate`.

Definition at line 180 of file `basic_ios.h`.

Referenced by `std::__detail::operator>>()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_istream< _CharT, _Traits >::sentry::sentry()`.

5.612.5.26 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::ignore (streamsize __n, int_type __delim) [inherited]`

Discarding characters.

Parameters

<code>__n</code>	Number of characters to discard.
<code>__delim</code>	A "stop" character.

Returns

*this

Extracts characters and throws them away until one of the following happens:

- if `__n != std::numeric_limits<int>::max()`, `__n` characters are extracted
- the input sequence reaches end-of-file
- the next character equals `__delim` (in this case, the character is extracted); note that this condition will never occur if `__delim` equals `traits::eof()`.

NB: Provide three overloads, instead of the single function (with defaults) mandated by the Standard: this leads to a better performing implementation, while still conforming to the Standard.

Definition at line 555 of file istream.tcc.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_streambuf< _CharT, _Traits >::sbumpc()`, `std::basic_ios< _CharT, _Traits >::setstate()`, `std::basic_streambuf< _CharT, _Traits >::sgetc()`, and `std::basic_streambuf< _CharT, _Traits >::snextc()`.

5.612.5.27 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::ignore (streamsize __n) [inherited]`

Simple extraction.

Returns

A character, or `eof()`.

Tries to extract a character. If none are available, sets failbit and returns `traits::eof()`.

Definition at line 493 of file istream.tcc.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_ios< _CharT, _Traits >::setstate()`, `std::basic_streambuf< _CharT, _Traits >::sgetc()`, and `std::basic_streambuf< _CharT, _Traits >::snextc()`.

5.612.5.28 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::ignore (void) [inherited]`

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns traits::eof().

Definition at line 460 of file istream.tcc.

References `std::basic_istream< _CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, `std::basic_streambuf< _CharT, _Traits>::sbumpc()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream< _CharT, _Traits>::getline()`, and `std::basic_istream< _CharT, _Traits>::ignore()`.

5.612.5.29 `template<typename _CharT, typename _Traits> locale std::basic_ios< _CharT, _Traits>::imbue (const locale & __loc) [inherited]`

Moves to a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls `ios_base::imbue(loc)`, and if a stream buffer is associated with this stream, calls that buffer's `pubimbue(loc)`.

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Definition at line 114 of file basic_ios.tcc.

References `std::ios_base::imbue()`.

Referenced by `std::basic_ios< char, _Traits>::fill()`.

5.612.5.30 `template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits>::init (basic_streambuf< _CharT, _Traits> * __sb) [protected], [inherited]`

All setup is performed here.

This is called from the public constructor. It is not virtual and cannot be redefined.

Definition at line 126 of file basic_ios.tcc.

Referenced by `std::basic_ios< char, _Traits>::basic_ios()`.

5.612.5.31 `template<typename _CharT, typename _Traits> bool std::basic_ifstream<_CharT, _Traits>::is_open ()`
`[inline]`

Wrapper to test for an open file.

Returns

`rdbuf() -> is_open()`

Definition at line 574 of file `fstream`.

5.612.5.32 `long& std::ios_base::iword (int __ix)` `[inline]`, `[inherited]`

Access to integer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to an integer associated with the index.

The `iword` function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 809 of file `ios_base.h`.

5.612.5.33 `template<typename _CharT, typename _Traits> char std::basic_ios<_CharT, _Traits>::narrow (char_type __c, char __default) const` `[inline]`, `[inherited]`

Squeezes characters.

Parameters

<code>__c</code>	The character to narrow.
<code>__default</code>	The character to narrow.

Returns

The narrowed character.

Maps a character of `char_type` to a character of `char`, if possible.

Returns the result of

```
1 std::use_facet<ctype<char_type> > (getloc()).narrow(c, default)
```

Additional l10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Definition at line 430 of file basic_ios.h.

```
5.612.5.34 template<typename _CharT, typename _Traits > void std::basic_ifstream<_CharT, _Traits >::open ( const char *
__s, ios_base::openmode __mode = ios_base::in ) [inline]
```

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Calls `std::basic_filebuf::open(s,__mode|in)`. If that function fails, `failbit` is set in the stream's error state.

Tip: When using `std::string` to hold the filename, you must use `.c_str()` before passing it to this constructor.

Definition at line 595 of file fstream.

```
5.612.5.35 template<typename _CharT, typename _Traits > void std::basic_ifstream<_CharT, _Traits >::open ( const
std::string & __s, ios_base::openmode __mode = ios_base::in ) [inline]
```

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Calls `std::basic_filebuf::open(__s,__mode|in)`. If that function fails, `failbit` is set in the stream's error state.

Definition at line 615 of file fstream.

```
5.612.5.36 template<typename _CharT, typename _Traits> std::basic_ios<_CharT, _Traits >::operator bool ( ) const
[inline], [explicit], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`

Definition at line 117 of file basic_ios.h.

```
5.612.5.37 template<typename _CharT, typename _Traits> bool std::basic_ios<_CharT, _Traits>::operator! ( ) const
[inline], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`.

Definition at line 125 of file `basic_ios.h`.

```
5.612.5.38 template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>
::operator>> ( __istream_type &(*)(__istream_type &)__pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

Definition at line 120 of file `istream`.

Referenced by `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::basic_istream<_CharT, _Traits>::sentry←::sentry()`, and `std::ws()`.

```
5.612.5.39 template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>
::operator>> ( __ios_type &(*)(__ios_type &)__pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

Definition at line 124 of file `istream`.

```
5.612.5.40 template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>
::operator>> ( ios_base &(*)(ios_base &)__pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

Definition at line 131 of file `istream`.

```
5.612.5.41 template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>
::operator>> ( bool &__n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

<code>_↔</code>	A variable of builtin integral type.
<code>_n</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 168 of file `istream`.

5.612.5.42 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::operator>> (short & __n) [inherited]`

Integer arithmetic extractors.

Parameters

<code>_↔</code>	A variable of builtin integral type.
<code>_n</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 114 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::failbit`, `std::num_get< _CharT, _InIter>::get()`, `std::ios_base::goodbit`, `std::basic_istream< _CharT, _Traits>::operator>>()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

5.612.5.43 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::operator>> (unsigned short & __n) [inline], [inherited]`

Integer arithmetic extractors.

Parameters

<code>_↔</code>	A variable of builtin integral type.
<code>_n</code>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 175 of file `istream`.

5.612.5.44 `template<typename _CharT, typename _Traits> basic_ifstream<_CharT, _Traits> & std::basic_ifstream<_CharT, _Traits>::operator>> (int &__n) [inherited]`

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 159 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::failbit`, `std::num_get<_CharT, _InIter>::get()`, `std::ios_base::goodbit`, `std::basic_ifstream<_CharT, _Traits>::operator>>()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

5.612.5.45 `template<typename _CharT, typename _Traits> __istream_type& std::basic_ifstream<_CharT, _Traits>::operator>> (unsigned int &__n) [inline], [inherited]`

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 182 of file `istream`.

5.612.5.46 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::operator>> (long & __n) [inline], [inherited]`

Integer arithmetic extractors.

Parameters

<code>_↔</code>	A variable of builtin integral type.
<code>_n</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 186 of file `istream`.

```
5.612.5.47 template<typename _CharT, typename _Traits> __istream_type& std::basic_ifstream<_CharT, _Traits>::operator>> ( unsigned long &_n ) [inline],[inherited]
```

Integer arithmetic extractors.

Parameters

<code>_↔</code>	A variable of builtin integral type.
<code>_n</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 190 of file `istream`.

```
5.612.5.48 template<typename _CharT, typename _Traits> __istream_type& std::basic_ifstream<_CharT, _Traits>::operator>> ( long long &_n ) [inline],[inherited]
```

Integer arithmetic extractors.

Parameters

<code>_↔</code>	A variable of builtin integral type.
<code>_n</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 195 of file `istream`.

5.612.5.49 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::operator>> (unsigned long long & __n) [inline], [inherited]`

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 199 of file `istream`.

5.612.5.50 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::operator>> (float & __f) [inline], [inherited]`

Floating point arithmetic extractors.

Parameters

<code>__f</code>	A variable of builtin floating point type.
------------------	--

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 214 of file `istream`.

5.612.5.51 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::operator>> (double & __f) [inline], [inherited]`

Floating point arithmetic extractors.

Parameters

\leftrightarrow	A variable of builtin floating point type.
$_ \leftrightarrow$	
\leftrightarrow	
$_ \leftrightarrow$	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 218 of file `istream`.

5.612.5.52 `template<typename _CharT, typename _Traits> __istream_type& std::basic_ifstream<_CharT, _Traits>::operator>> (long double &__f) [inline], [inherited]`

Floating point arithmetic extractors.

Parameters

\leftrightarrow	A variable of builtin floating point type.
$_ \leftrightarrow$	
\leftrightarrow	
$_ \leftrightarrow$	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 222 of file `istream`.

5.612.5.53 `template<typename _CharT, typename _Traits> __istream_type& std::basic_ifstream<_CharT, _Traits>::operator>> (void *&__p) [inline], [inherited]`

Basic arithmetic extractors.

Parameters

$_ \leftrightarrow$	A variable of pointer type.
<i>p</i>	

Returns

*`this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 235 of file `istream`.

5.612.5.54 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::operator>> (__streambuf_type * __sb) [inherited]`

Extracting into another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is `NULL`, the stream will set `failbit` in its error state.

Characters are extracted from this stream and inserted into the `__sb` streambuf until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs (and in this case is caught)

If the function inserts no characters, `failbit` is set.

Definition at line 204 of file `istream.tcc`.

References `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::basic_istream<_CharT, _Traits>::get()`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

5.612.5.55 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits>::int_type std::basic_istream<_CharT, _Traits>::peek (void) [inherited]`

Looking ahead in the stream.

Returns

The next character, or `eof()`.

If, after constructing the sentry object, `good()` is `false`, returns `traits::eof()`. Otherwise reads but does not extract the next input character.

Definition at line 620 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_istream<_CharT, _Traits>::read()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream<_CharT, _Traits>::ignore()`.

5.612.5.56 **streamsize** std::ios_base::precision () const [inline],[inherited]

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Definition at line 689 of file ios_base.h.

Referenced by std::basic_ios<_CharT, _Traits>::copyfmt(), std::num_get<_CharT, _InIter>::do_get(), std::normal_↵distribution<_RealType>::operator>(), std::gamma_distribution<_RealType>::operator>(), std::negative_binomial_↵distribution<_IntType>::operator>(), and std::operator>>().

5.612.5.57 **streamsize** std::ios_base::precision (streamsize __prec) [inline],[inherited]

Changing flags.

Parameters

<code>__prec</code>	The new precision value.
---------------------	--------------------------

Returns

The previous value of precision().

Definition at line 698 of file ios_base.h.

5.612.5.58 **template**<typename _CharT, typename _Traits> **basic_istream**<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::putback (char_type __c) [inherited]

Unextracting a single character.

Parameters

<code>__↵ __c</code>	The character to push back into the input stream.
--------------------------	---

Returns

*this

If rdbuf () is not null, calls rdbuf ()->sputbackc (c) .

If rdbuf () is null or if sputbackc () fails, sets badbit in the error state.

Note

This function first clears eofbit. Since no characters are extracted, the next call to `gcount()` will return 0, as required by DR 60.

Definition at line 711 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::rdstate()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sputbackc()`, and `std::basic_istream<_CharT, _Traits>::unget()`.

Referenced by `std::basic_istream<_CharT, _Traits>::readsome()`.

5.612.5.59 `void*& std::ios_base::pword (int __ix)` `[inline]`, `[inherited]`

Access to void pointer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to a `void*` associated with the index.

The `pword` function provides access to an array of pointers that can be used for any purpose. The array grows as required to hold the supplied index. All pointers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 830 of file `ios_base.h`.

5.612.5.60 `template<typename _CharT, typename _Traits> basic_streambuf<_CharT, _Traits> * std::basic_ios<_CharT, _Traits>::rdbuf (basic_streambuf<_CharT, _Traits> * __sb)` `[inherited]`

Changing the underlying buffer.

Parameters

<code>__sb</code>	The new stream buffer.
-------------------	------------------------

Returns

The previous stream buffer.

Associates a new buffer with the current stream, and clears the error state.

Due to historical accidents which the LWG refuses to correct, the I/O library suffers from a design error: this function is hidden in derived classes by overrides of the zero-argument `rdbuf()`, which is non-virtual for hysterical raisins. As a result, you must use explicit qualifications to access this function via any derived class. For example:

```
1 std::fstream    foo;           // or some other derived type
2 std::streambuf* p = .....;
3
4 foo.ios::rdbuf(p);           // ios == basic_ios<char>
```

Definition at line 53 of file `basic_ios.tcc`.

5.612.5.61 `template<typename _CharT, typename _Traits> __filebuf_type* std::basic_ifstream<_CharT, _Traits>::rdbuf () const [inline]`

Accessing the underlying buffer.

Returns

The current `basic_filebuf` buffer.

This hides both signatures of `std::basic_ios::rdbuf()`.

Definition at line 566 of file `fstream`.

5.612.5.62 `template<typename _CharT, typename _Traits> iosstate std::basic_ios<_CharT, _Traits>::rdstate () const [inline], [inherited]`

Returns the error state of the stream buffer.

Returns

A bit pattern (well, isn't everything?)

See `std::ios_base::iosstate` for the possible bit values. Most users will call one of the interpreting wrappers, e.g., `good()`.

Definition at line 137 of file `basic_ios.h`.

Referenced by `std::basic_ios<char, _Traits>::bad()`, `std::basic_ios<char, _Traits>::eof()`, `std::basic_ios<char, _Traits>::fail()`, `std::basic_ios<char, _Traits>::good()`, `std::basic_istream<_CharT, _Traits>::putback()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_ios<char, _Traits>::setstate()`, and `std::basic_istream<_CharT, _Traits>::unget()`.

5.612.5.63 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::read (char_type * __s, streamsize __n) [inherited]`

Extraction without delimiters.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

`*this`

If the stream state is `good()`, extracts characters and stores them into `__s` until one of the following happens:

- `__n` characters are stored
- the input sequence reaches end-of-file, in which case the error state is set to `failbit|eofbit`.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 650 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_istream<_CharT, _Traits>::readsome()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream<_CharT, _Traits>::peek()`.

5.612.5.64 `template<typename _CharT, typename _Traits> streamsize std::basic_istream<_CharT, _Traits>::readsome (char_type* __s, streamsize __n) [inherited]`

Extraction until the buffer is exhausted, but no more.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

The number of characters extracted.

Extracts characters and stores them into `__s` depending on the number of characters remaining in the streambuf's buffer, `rdbuf()->in_avail()`, called A here:

- if $A == -1$, sets eofbit and extracts no characters
- if $A == 0$, extracts no characters
- if $A > 0$, extracts $\min(A, n)$

The goal is to empty the current buffer, and to not request any more from the external input sequence controlled by the streambuf.

Definition at line 679 of file istream.tcc.

References `std::basic_istream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::min()`, `std::basic_istream<_CharT, _Traits>::putback()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream<_CharT, _Traits>::read()`.

5.612.5.65 `void std::ios_base::register_callback (event_callback __fn, int __index)` [inherited]

Add the callback `__fn` with parameter `__index`.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

5.612.5.66 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::seekg (pos_type __pos)` [inherited]

Changing the current read position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

*this

If `fail()` is not true, calls `rdbuf()->pubseekpos(__pos)`. If that function fails, sets failbit.

Note

This function first clears eofbit. It does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

Definition at line 845 of file istream.tcc.

References `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::ios_base::in`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::rdstate()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream<_CharT, _Traits>::tellg()`.

5.612.5.67 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::seekg (off_type __off, ios_base::seekdir __dir) [inherited]`

Changing the current read position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

`*this`

If `fail()` is not true, calls `rdbuf()->pubseekoff(__off, __dir)`. If that function fails, sets failbit.

Note

This function first clears eofbit. It does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

Definition at line 884 of file istream.tcc.

References `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::ios_base::in`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::rdstate()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

5.612.5.68 `fmtflags std::ios_base::setf (fmtflags __fmtfl) [inline], [inherited]`

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
----------------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set.

Definition at line 646 of file ios_base.h.

Referenced by std::boolalpha(), std::dec(), std::fixed(), std::hex(), std::hexfloat(), std::left(), std::oct(), std::__detail::operator>>(), std::right(), std::scientific(), std::showbase(), std::showpoint(), std::showpos(), std::skipws(), std::unitbuf(), and std::uppercase().

5.612.5.69 `fmtflags std::ios_base::setf (fmtflags __fmtfl, fmtflags __mask)` `[inline], [inherited]`

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
<code>__mask</code>	The flags mask for <i>fmtfl</i> .

Returns

The previous format control flags.

This function clears *mask* in the format flags, then sets *fmtfl* & *mask*. An example mask is `ios_base::adjustfield`.

Definition at line 663 of file ios_base.h.

5.612.5.70 `template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits >::setstate (iostate __state)` `[inline], [inherited]`

Sets additional flags in the error state.

Parameters

<code>__state</code>	The additional state flag(s) to set.
----------------------	--------------------------------------

See `std::ios_base::iostate` for the possible bit values.

Definition at line 157 of file basic_ios.h.

Referenced by std::basic_ostream< _CharT, _Traits >::flush(), std::basic_istream< _CharT, _Traits >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::getline(), std::basic_istream< _CharT, _Traits >::ignore(), std::basic_ostream< _CharT, _Traits >::operator<<(), std::operator<<(), std::basic_istream< _CharT, _Traits >::operator>>(), std::tr2::operator>>(), std::operator>>(), std::basic_istream< _CharT, _Traits >::peek(), std::basic_ostream< _CharT, _Traits >::put(), std::basic_istream< _CharT, _Traits >::putback(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::readsome(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_ostream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::sync(), std::basic_istream< _CharT, _Traits >::unget(), and std::ws().

5.612.5.71 `template<typename _CharT, typename _Traits> int std::basic_istream<_CharT, _Traits>::sync (void)`
`[inherited]`

Synchronizing the stream buffer.

Returns

0 on success, -1 on failure

If `rdbuf()` is a null pointer, returns -1.

Otherwise, calls `rdbuf()->pubsync()`, and if that returns -1, sets `badbit` and returns -1.

Otherwise, returns 0.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

Definition at line 781 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::goodbit`, `std::basic_streambuf<_CharT, _Traits>::pubsync()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::setstate()`, and `std::basic_istream<_CharT, _Traits>::tellg()`.

Referenced by `std::basic_istream<_CharT, _Traits>::unget()`.

5.612.5.72 `static bool std::ios_base::sync_with_stdio (bool __sync = true)` `[static]`, `[inherited]`

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., `stdout`) and the standard C++ objects (e.g., `cout`). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>

5.612.5.73 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits>::pos_type`
`std::basic_istream<_CharT, _Traits>::tellg (void)` `[inherited]`

Getting the current read position.

Returns

A file position object.

If `fail()` is not false, returns `pos_type(-1)` to indicate failure. Otherwise returns `rdbuf() -> pubseekoff(0, cur, in)`.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`. At variance with `putback`, `unget` and `seekg`, `eofbit` is not cleared first.

Definition at line 817 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::cur`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::ios_base::in`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, and `std::basic_ifstream< _CharT, _Traits >::seekg()`.

Referenced by `std::basic_ifstream< _CharT, _Traits >::sync()`.

5.612.5.74 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits>* std::basic_ios< _CharT, _Traits>::tie() const` `[inline]`, `[inherited]`

Fetches the current *tied* stream.

Returns

A pointer to the tied stream, or NULL if the stream is not tied.

A stream may be *tied* (or synchronized) to a second output stream. When this stream performs any I/O, the tied stream is first flushed. For example, `std::cin` is tied to `std::cout`.

Definition at line 295 of file `basic_ios.h`.

Referenced by `std::basic_ios< char, _Traits >::basic_ios()`, `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_ifstream< _CharT, _Traits >::sentry::sentry()`.

5.612.5.75 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits>* std::basic_ios< _CharT, _Traits>::tie(basic_ostream< _CharT, _Traits>* __tiestr)` `[inline]`, `[inherited]`

Ties this stream to an output stream.

Parameters

<code>__tiestr</code>	The output stream.
-----------------------	--------------------

Returns

The previously tied output stream, or NULL if the stream was not tied.

This sets up a new tie; see `tie()` for more.

Definition at line 307 of file `basic_ios.h`.

5.612.5.76 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::unget (void) [inherited]`

Unextracting the previous character.

Returns

*this

If `rdbuf()` is not null, calls `rdbuf()->sungetc(c)`.

If `rdbuf()` is null or if `sungetc()` fails, sets `badbit` in the error state.

Note

This function first clears `eofbit`. Since no characters are extracted, the next call to `gcount()` will return 0, as required by DR 60.

Definition at line 746 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::rdstate()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sungetc()`, and `std::basic_istream<_CharT, _Traits>::sync()`.

Referenced by `std::__detail::operator>>()`, and `std::basic_istream<_CharT, _Traits>::putback()`.

5.612.5.77 `void std::ios_base::unsetf (fmtflags __mask) [inline], [inherited]`

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

Definition at line 678 of file `ios_base.h`.

Referenced by `std::defaultfloat()`, `std::noboolalpha()`, `std::noshowbase()`, `std::noshowpoint()`, `std::noshowpos()`, `std::noskipws()`, `std::nounitbuf()`, and `std::nouppercase()`.

5.612.5.78 `template<typename _CharT, typename _Traits> char_type std::basic_ios<_CharT, _Traits>::widen (char __c) const [inline], [inherited]`

Widens characters.

Parameters

<code>_↔</code>	The character to widen.
<code>_c</code>	

Returns

The widened character.

Maps a character of `char` to a character of `char_type`.

Returns the result of

```
1 std::use_facet<ctype<char_type>> >(getloc()).widen(c)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.↔html>

Definition at line 449 of file `basic_ios.h`.

Referenced by `std::basic_ios<char, _Traits>::fill()`, `std::getline()`, `std::normal_distribution<_RealType>::operator()()`, `std::gamma_distribution<_RealType>::operator()()`, `std::negative_binomial_distribution<_IntType>::operator()()`, `std::tr2::operator>>()`, and `std::operator>>()`.

5.612.5.79 `streamsize std::ios_base::width() const` `[inline]`, `[inherited]`

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Definition at line 712 of file `ios_base.h`.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::money_get<_CharT, _Inlter>::do_get()`, `std::num_↔_get<_CharT, _Inlter>::do_get()`, `std::num_put<_CharT, _Outlter>::do_put()`, and `std::operator>>()`.

5.612.5.80 `streamsize std::ios_base::width(streamsize __wide)` `[inline]`, `[inherited]`

Changing flags.

Parameters

<code>__wide</code>	The new width value.
---------------------	----------------------

Returns

The previous value of `width()`.

Definition at line 721 of file `ios_base.h`.

5.612.5.81 `static int std::ios_base::xalloc () throw` `[static], [inherited]`

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the `iword` and `pword` functions. The expectation is that an application calls `xalloc` in order to obtain an index in the `iword` and `pword` arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. `xalloc` is guaranteed to return an index that is safe to use in the `iword` and `pword` arrays.

5.612.6 Member Data Documentation

5.612.6.1 `template<typename _CharT, typename _Traits> streamsize std::basic_istream< _CharT, _Traits >::_M_gcount`
`[protected], [inherited]`

The number of characters extracted in the previous unformatted function; see `gcount()`.

Definition at line 82 of file `istream`.

Referenced by `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

5.612.6.2 `const fmtflags std::ios_base::adjustfield` `[static], [inherited]`

A mask of `left|right|internal`. Useful for the 2-arg form of `setf`.

Definition at line 378 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::internal()`, `std::left()`, and `std::right()`.

5.612.6.3 `const openmode std::ios_base::app` `[static], [inherited]`

Seek to end before each write.

Definition at line 432 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

5.612.6.4 const openmode std::ios_base::ate [static],[inherited]

Open and seek to end immediately after opening.

Definition at line 435 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::open().

5.612.6.5 const iostate std::ios_base::badbit [static],[inherited]

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Definition at line 402 of file ios_base.h.

Referenced by std::basic_ios< char, _Traits >::bad(), std::basic_ios< char, _Traits >::fail(), std::basic_ostream< _CharT, _Traits >::flush(), std::basic_istream< _CharT, _Traits >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::basic_istream< _CharT, _Traits >::ignore(), std::basic_ostream< _CharT, _Traits >::operator<<(), std::operator<<(), std::basic_istream< _CharT, _Traits >::operator>>(), std::operator>>(), std::basic_istream< _CharT, _Traits >::peek(), std::basic_ostream< _CharT, _Traits >::put(), std::basic_istream< _CharT, _Traits >::putback(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::readsome(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_ostream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::sync(), std::basic_istream< _CharT, _Traits >::tellg(), std::basic_ostream< _CharT, _Traits >::tellp(), std::basic_istream< _CharT, _Traits >::unget(), and std::basic_ostream< _CharT, _Traits >::write().

5.612.6.6 const fmtflags std::ios_base::basefield [static],[inherited]

A mask of dec|oct|hex. Useful for the 2-arg form of setf.

Definition at line 381 of file ios_base.h.

Referenced by std::dec(), std::num_get< _CharT, _InIter >::do_get(), std::num_put< _CharT, _OutIter >::do_put(), std::hex(), std::oct(), and std::basic_ostream< _CharT, _Traits >::operator<<().

5.612.6.7 const seekdir std::ios_base::beg [static],[inherited]

Request a seek relative to the beginning of the stream.

Definition at line 464 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::seekpos(), and __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync().

5.612.6.8 const openmode std::ios_base::binary [static],[inherited]

Perform input and output in binary mode (as opposed to text mode). This is probably not what you think it is; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.binary>.

Definition at line 440 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::showmanyc().

5.612.6.9 `const fmtflags std::ios_base::boolalpha` `[static], [inherited]`

Insert/extract `bool` in alphabetic rather than numeric format.

Definition at line 326 of file `ios_base.h`.

Referenced by `std::boolalpha()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, and `std::noboolalpha()`.

5.612.6.10 `const seekdir std::ios_base::cur` `[static], [inherited]`

Request a seek relative to the current position within the sequence.

Definition at line 467 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::imbue()`, `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`, `std::basic_filebuf< _CharT, _Traits >::seekoff()`, `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::tellg()`, and `std::basic_ostream< _CharT, _Traits >::tellp()`.

5.612.6.11 `const fmtflags std::ios_base::dec` `[static], [inherited]`

Converts integer input or generates integer output in decimal base.

Definition at line 329 of file `ios_base.h`.

Referenced by `std::dec()`.

5.612.6.12 `const seekdir std::ios_base::end` `[static], [inherited]`

Request a seek relative to the current end of the sequence.

Definition at line 470 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::open()`, and `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`.

5.612.6.13 `const iostate std::ios_base::eofbit` `[static], [inherited]`

Indicates that an input operation reached the end of an input sequence.

Definition at line 405 of file `ios_base.h`.

Referenced by `std::time_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::time_get< _CharT, _InIter >::do_get_date()`, `std::time_get< _CharT, _InIter >::do_get_monthname()`, `std::time_get< _CharT, _InIter >::do_get_time()`, `std::time_get< _CharT, _InIter >::do_get_weekday()`, `std::time_get< _CharT, _InIter >::do_get_year()`, `std::basic_ios< char, _Traits >::eof()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::time_get< _CharT, _InIter >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::ws()`.

5.612.6.14 const iostate std::ios_base::failbit [static], [inherited]

Indicates that an input operation failed to read the expected characters, or that an output operation failed to generate the desired characters.

Definition at line 410 of file ios_base.h.

Referenced by std::time_get< _CharT, _Inlter >::do_date_order(), std::num_get< _CharT, _Inlter >::do_get(), std::time_get< _CharT, _Inlter >::do_get_monthname(), std::time_get< _CharT, _Inlter >::do_get_weekday(), std::time_get< _CharT, _Inlter >::do_get_year(), std::basic_ios< char, _Traits >::fail(), std::basic_istream< _CharT, _Traits >::get(), std::time_get< _CharT, _Inlter >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::basic_ostream< _CharT, _Traits >::operator<<(), std::basic_istream< _CharT, _Traits >::operator>>(), std::operator>>(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_ostream< _CharT, _Traits >::sentry::sentry(), and std::basic_istream< _CharT, _Traits >::sentry::sentry().

5.612.6.15 const fmtflags std::ios_base::fixed [static], [inherited]

Generate floating-point output in fixed-point notation.

Definition at line 332 of file ios_base.h.

Referenced by std::num_get< _CharT, _Inlter >::do_get(), std::fixed(), and std::hexfloat().

5.612.6.16 const fmtflags std::ios_base::floatfield [static], [inherited]

A mask of scientific|fixed. Useful for the 2-arg form of setf.

Definition at line 384 of file ios_base.h.

Referenced by std::defaultfloat(), std::num_get< _CharT, _Inlter >::do_get(), std::fixed(), std::hexfloat(), and std::scientific().

5.612.6.17 const iostate std::ios_base::goodbit [static], [inherited]

Indicates all is well.

Definition at line 413 of file ios_base.h.

Referenced by std::time_get< _CharT, _Inlter >::do_date_order(), std::time_get< _CharT, _Inlter >::do_get(), std::num_get< _CharT, _Inlter >::do_get(), std::time_get< _CharT, _Inlter >::do_get_monthname(), std::time_get< _CharT, _Inlter >::do_get_weekday(), std::time_get< _CharT, _Inlter >::do_get_year(), std::basic_ostream< _CharT, _Traits >::flush(), std::basic_istream< _CharT, _Traits >::get(), std::time_get< _CharT, _Inlter >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::basic_istream< _CharT, _Traits >::ignore(), std::basic_ostream< _CharT, _Traits >::operator<<(), std::basic_istream< _CharT, _Traits >::operator>>(), std::operator>>(), std::basic_istream< _CharT, _Traits >::peek(), std::basic_ostream< _CharT, _Traits >::put(), std::basic_istream< _CharT, _Traits >::putback(), std::basic_ios< char, _Traits >::rdstate(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::readsomewhat(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_ostream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::sync(), and std::basic_istream< _CharT, _Traits >::unset().

5.612.6.18 const fmtflags std::ios_base::hex [static],[inherited]

Converts integer input or generates integer output in hexadecimal base.

Definition at line 335 of file ios_base.h.

Referenced by std::num_get< _CharT, _Inlter >::do_get(), std::num_put< _CharT, _Outlter >::do_put(), std::hex(), and std::basic_ostream< _CharT, _Traits >::operator<<().

5.612.6.19 const openmode std::ios_base::in [static],[inherited]

Open for input. Default for ifstream and fstream.

Definition at line 443 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::pbackfail(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos(), std::basic_filebuf< _CharT, _Traits >::showmanyc(), __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync(), std::basic_istream< _CharT, _Traits >::tellg(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::underflow(), std::basic_filebuf< _CharT, _Traits >::underflow(), and std::basic_filebuf< _CharT, _Traits >::xsgetn().

5.612.6.20 const fmtflags std::ios_base::internal [static],[inherited]

Adds fill characters at a designated internal point in certain generated output, or identical to right if no such point is designated.

Definition at line 340 of file ios_base.h.

Referenced by std::money_get< _CharT, _Inlter >::do_get(), std::num_put< _CharT, _Outlter >::do_put(), and std::internal().

5.612.6.21 const fmtflags std::ios_base::left [static],[inherited]

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.)

Definition at line 344 of file ios_base.h.

Referenced by std::money_get< _CharT, _Inlter >::do_get(), std::num_put< _CharT, _Outlter >::do_put(), and std::left().

5.612.6.22 const fmtflags std::ios_base::oct [static],[inherited]

Converts integer input or generates integer output in octal base.

Definition at line 347 of file ios_base.h.

Referenced by std::num_get< _CharT, _Inlter >::do_get(), std::oct(), and std::basic_ostream< _CharT, _Traits >::operator<<().

5.612.6.23 const openmode std::ios_base::out [static],[inherited]

Open for output. Default for ofstream and fstream.

Definition at line 446 of file ios_base.h.

Referenced by std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow(), std::basic_filebuf< _CharT, _Traits >::overflow(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::pbackfail(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos(), __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync(), std::basic_ostream< _CharT, _Traits >::tellp(), and std::basic_filebuf< _CharT, _Traits >::xsputn().

5.612.6.24 const fmtflags std::ios_base::right [static],[inherited]

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Definition at line 351 of file ios_base.h.

Referenced by std::right().

5.612.6.25 const fmtflags std::ios_base::scientific [static],[inherited]

Generates floating-point output in scientific notation.

Definition at line 354 of file ios_base.h.

Referenced by std::hexfloat(), and std::scientific().

5.612.6.26 const fmtflags std::ios_base::showbase [static],[inherited]

Generates a prefix indicating the numeric base of generated integer output.

Definition at line 358 of file ios_base.h.

Referenced by std::money_get< _CharT, _InIter >::do_get(), std::num_get< _CharT, _InIter >::do_get(), std::num_put< _CharT, _OutIter >::do_put(), std::noshowbase(), and std::showbase().

5.612.6.27 const fmtflags std::ios_base::showpoint [static],[inherited]

Generates a decimal-point character unconditionally in generated floating-point output.

Definition at line 362 of file ios_base.h.

Referenced by std::noshowpoint(), and std::showpoint().

5.612.6.28 const fmtflags std::ios_base::showpos [static],[inherited]

Generates a + sign in non-negative generated numeric output.

Definition at line 365 of file ios_base.h.

Referenced by std::num_get< _CharT, _InIter >::do_get(), std::noshowpos(), and std::showpos().

Public Types

- enum [event](#) { [erase_event](#), [imbue_event](#), [copyfmt_event](#) }
 - typedef void(* [event_callback](#)) ([event](#) __e, [ios_base](#) &__b, int __i)
 - typedef _ios_Fmtflags [fmtflags](#)
 - typedef int [io_state](#)
 - typedef _ios_istate [iostate](#)
 - typedef int [open_mode](#)
 - typedef _ios_Openmode [openmode](#)
 - typedef int [seek_dir](#)
 - typedef _ios_Seekdir [seekdir](#)
 - typedef [std::streamoff](#) [streamoff](#)
 - typedef [std::streampos](#) [streampos](#)
-
- typedef _CharT [char_type](#)
 - typedef _Traits::int_type [int_type](#)
 - typedef _Traits::pos_type [pos_type](#)
 - typedef _Traits::off_type [off_type](#)
 - typedef _Traits [traits_type](#)
-
- typedef [ctype](#)< _CharT > [__ctype_type](#)
 - typedef [num_put](#)< _CharT, [ostreambuf_iterator](#)< _CharT, _Traits > > [__num_put_type](#)
 - typedef [num_get](#)< _CharT, [istreambuf_iterator](#)< _CharT, _Traits > > [__num_get_type](#)

Public Member Functions

- [basic_ios](#) ([basic_streambuf](#)< _CharT, _Traits > *__sb)
- virtual [~basic_ios](#) ()
- const [locale](#) & [_M_getloc](#) () const
- void [_M_setstate](#) ([iostate](#) __state)
- bool [bad](#) () const
- void [clear](#) ([iostate](#) __state=[goodbit](#))
- [basic_ios](#) & [copyfmt](#) (const [basic_ios](#) &__rhs)
- bool [eof](#) () const
- [iostate](#) [exceptions](#) () const
- void [exceptions](#) ([iostate](#) __except)
- bool [fail](#) () const
- [char_type](#) [fill](#) () const
- [char_type](#) [fill](#) ([char_type](#) __ch)
- [fmtflags](#) [flags](#) () const
- [fmtflags](#) [flags](#) ([fmtflags](#) __fmtfl)
- [locale](#) [getloc](#) () const
- bool [good](#) () const
- [locale](#) [imbue](#) (const [locale](#) &__loc)
- long & [iword](#) (int __ix)
- char [narrow](#) ([char_type](#) __c, char __dfault) const
- [streamsize](#) [precision](#) () const
- [streamsize](#) [precision](#) ([streamsize](#) __prec)
- void *& [pword](#) (int __ix)

- [basic_streambuf](#)< [_CharT](#), [_Traits](#) > * [rdbuf](#) () const
 - [basic_streambuf](#)< [_CharT](#), [_Traits](#) > * [rdbuf](#) ([basic_streambuf](#)< [_CharT](#), [_Traits](#) > * __sb)
 - [iosstate](#) [rdstate](#) () const
 - void [register_callback](#) ([event_callback](#) __fn, int __index)
 - [fmtflags](#) [setf](#) ([fmtflags](#) __fmtfl)
 - [fmtflags](#) [setf](#) ([fmtflags](#) __fmtfl, [fmtflags](#) __mask)
 - void [setstate](#) ([iosstate](#) __state)
 - [basic_ostream](#)< [_CharT](#), [_Traits](#) > * [tie](#) () const
 - [basic_ostream](#)< [_CharT](#), [_Traits](#) > * [tie](#) ([basic_ostream](#)< [_CharT](#), [_Traits](#) > * __tiestr)
 - void [unsetf](#) ([fmtflags](#) __mask)
 - [char_type](#) [widen](#) (char __c) const
 - [streamsize](#) [width](#) () const
 - [streamsize](#) [width](#) ([streamsize](#) __wide)
-
- [operator bool](#) () const
 - bool [operator!](#) () const

Static Public Member Functions

- static bool [sync_with_stdio](#) (bool __sync=true)
- static int [xalloc](#) () throw ()

Static Public Attributes

- static const [fmtflags](#) [adjustfield](#)
- static const [openmode](#) [app](#)
- static const [openmode](#) [ate](#)
- static const [iosstate](#) [badbit](#)
- static const [fmtflags](#) [basefield](#)
- static const [seekdir](#) [beg](#)
- static const [openmode](#) [binary](#)
- static const [fmtflags](#) [boolalpha](#)
- static const [seekdir](#) [cur](#)
- static const [fmtflags](#) [dec](#)
- static const [seekdir](#) [end](#)
- static const [iosstate](#) [eofbit](#)
- static const [iosstate](#) [failbit](#)
- static const [fmtflags](#) [fixed](#)
- static const [fmtflags](#) [floatfield](#)
- static const [iosstate](#) [goodbit](#)
- static const [fmtflags](#) [hex](#)
- static const [openmode](#) [in](#)
- static const [fmtflags](#) [internal](#)
- static const [fmtflags](#) [left](#)
- static const [fmtflags](#) [oct](#)
- static const [openmode](#) [out](#)
- static const [fmtflags](#) [right](#)
- static const [fmtflags](#) [scientific](#)

- static const [fmtflags](#) showbase
- static const [fmtflags](#) showpoint
- static const [fmtflags](#) showpos
- static const [fmtflags](#) skipws
- static const [openmode](#) trunc
- static const [fmtflags](#) unitbuf
- static const [fmtflags](#) uppercase

Protected Types

- enum { **_S_local_word_size** }

Protected Member Functions

- [basic_ios](#) ()
- **basic_ios** (const [basic_ios](#) &)=delete
- void **_M_cache_locale** (const [locale](#) &__loc)
- void **_M_call_callbacks** ([event](#) __ev) throw ()
- void **_M_dispose_callbacks** (void) throw ()
- [_Words](#) & **_M_grow_words** (int __index, bool __iword)
- void **_M_init** () throw ()
- void **_M_move** ([ios_base](#) &) noexcept
- void **_M_swap** ([ios_base](#) &__rhs) noexcept
- void **init** ([basic_streambuf](#)< [_CharT](#), [_Traits](#) > *__sb)
- void **move** ([basic_ios](#) &__rhs)
- void **move** ([basic_ios](#) &&__rhs)
- **basic_ios** & **operator=** (const [basic_ios](#) &)=delete
- void **set_rdbuf** ([basic_streambuf](#)< [_CharT](#), [_Traits](#) > *__sb)
- void **swap** ([basic_ios](#) &__rhs) noexcept

Protected Attributes

- [_Callback_list](#) * **_M_callbacks**
- const [__ctype_type](#) * **_M_ctype**
- [iostate](#) **_M_exception**
- [char_type](#) **_M_fill**
- bool **_M_fill_init**
- [fmtflags](#) **_M_flags**
- [locale](#) **_M_ios_locale**
- [_Words](#) **_M_local_word** [[_S_local_word_size](#)]
- const [__num_get_type](#) * **_M_num_get**
- const [__num_put_type](#) * **_M_num_put**
- [streamsize](#) **_M_precision**
- [basic_streambuf](#)< [_CharT](#), [_Traits](#) > * **_M_streambuf**
- [iostate](#) **_M_streambuf_state**
- [basic_ostream](#)< [_CharT](#), [_Traits](#) > * **_M_tie**
- [streamsize](#) **_M_width**
- [_Words](#) * **_M_word**
- int **_M_word_size**
- [_Words](#) **_M_word_zero**

5.613.1 Detailed Description

```
template<typename _CharT, typename _Traits>
class std::basic_ios< _CharT, _Traits >
```

Template class basic_ios, virtual base class for all stream classes.

Template Parameters

<code>_CharT</code>	Type of character stream.
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .

Most of the member functions called dispatched on stream objects (e.g., `std::cout.foo(bar)`;) are consolidated in this class.

Definition at line 77 of file `iosfwd`.

5.613.2 Member Typedef Documentation

5.613.2.1 `template<typename _CharT, typename _Traits> typedef ctype<_CharT> std::basic_ios< _CharT, _Traits >::__ctype_type`

These are non-standard types.

Definition at line 87 of file `basic_ios.h`.

5.613.2.2 `template<typename _CharT, typename _Traits> typedef num_get<_CharT, istreambuf_iterator<_CharT, _Traits> > std::basic_ios< _CharT, _Traits >::__num_get_type`

These are non-standard types.

Definition at line 91 of file `basic_ios.h`.

5.613.2.3 `template<typename _CharT, typename _Traits> typedef num_put<_CharT, ostreambuf_iterator<_CharT, _Traits> > std::basic_ios< _CharT, _Traits >::__num_put_type`

These are non-standard types.

Definition at line 89 of file `basic_ios.h`.

5.613.2.4 `template<typename _CharT, typename _Traits> typedef _CharT std::basic_ios< _CharT, _Traits >::__char_type`




These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

Definition at line 76 of file `basic_ios.h`.

5.613.2.5 `typedef void(* std::ios_base::event_callback)(event __e, ios_base &__b, int __i) [inherited]`

The type of an event callback function.

Parameters

 _e	One of the members of the event enum.
 _b	Reference to the ios_base object.
 _i	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several ios_base and basic_ios functions, specifically imbue(), copyfmt(), and ~ios().

Definition at line 504 of file ios_base.h.

5.613.2.6 typedef _Ios_Fmtflags std::ios_base::fmtflags [inherited]

This is a bitmask type.

_Ios_Fmtflags is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type fmtflags are:

- boolalpha
- dec
- fixed
- hex
- internal
- left
- oct
- right
- scientific
- showbase
- showpoint
- showpos
- skipws
- unitbuf
- uppercase
- adjustfield
- basefield
- floatfield

Definition at line 323 of file ios_base.h.

5.613.2.7 `template<typename _CharT, typename _Traits> typedef _Traits::int_type std::basic_ios<_CharT, _Traits>::int_type`

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

Definition at line 77 of file `basic_ios.h`.

5.613.2.8 `typedef _ios_iostate std::ios_base::iostate` `[inherited]`

This is a bitmask type.

`_Ios_Iostate` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `iostate` are:

- `badbit`
- `eofbit`
- `failbit`
- `goodbit`

Definition at line 398 of file `ios_base.h`.

5.613.2.9 `template<typename _CharT, typename _Traits> typedef _Traits::off_type std::basic_ios<_CharT, _Traits>::off_type`

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

Definition at line 79 of file `basic_ios.h`.

5.613.2.10 `typedef _ios_Openmode std::ios_base::openmode` `[inherited]`

This is a bitmask type.

`_Ios_Openmode` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `openmode` are:

- `app`
- `ate`
- `binary`
- `in`
- `out`
- `trunc`

Definition at line 429 of file `ios_base.h`.

5.613.2.11 `template<typename _CharT, typename _Traits> typedef _Traits::pos_type std::basic_ios< _CharT, _Traits >::pos_type`

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

Definition at line 78 of file `basic_ios.h`.

5.613.2.12 `typedef _Ios_Seekdir std::ios_base::seekdir` `[inherited]`

This is an enumerated type.

`_Ios_Seekdir` is implementation-defined. Defined values of type `seekdir` are:

- `beg`
- `cur`, equivalent to `SEEK_CUR` in the C standard library.
- `end`, equivalent to `SEEK_END` in the C standard library.

Definition at line 461 of file `ios_base.h`.

5.613.2.13 `template<typename _CharT, typename _Traits> typedef _Traits std::basic_ios< _CharT, _Traits >::traits_type`

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

Definition at line 80 of file `basic_ios.h`.

5.613.3 Member Enumeration Documentation

5.613.3.1 `enum std::ios_base::event` `[inherited]`

The set of events that may be passed to an event callback.

`erase_event` is used during `~ios()` and `copyfmt()`. `imbue_event` is used during `imbue()`. `copyfmt_event` is used during `copyfmt()`.

Definition at line 487 of file `ios_base.h`.

5.613.4 Constructor & Destructor Documentation

5.613.4.1 `template<typename _CharT, typename _Traits> std::basic_ios< _CharT, _Traits >::basic_ios (basic_streambuf< _CharT, _Traits > * __sb)` `[inline]`, `[explicit]`

Constructor performs initialization.

The parameter is passed by derived streams.

Definition at line 270 of file `basic_ios.h`.

5.613.4.2 `template<typename _CharT, typename _Traits> virtual std::basic_ios< _CharT, _Traits >::~basic_ios ()`
`[inline], [virtual]`

Empty.

The destructor does nothing. More specifically, it does not destroy the streambuf held by rdbuf().

Definition at line 282 of file basic_ios.h.

5.613.4.3 `template<typename _CharT, typename _Traits> std::basic_ios< _CharT, _Traits >::basic_ios ()` `[inline],`
`[protected]`

Empty.

The default constructor does nothing and is not normally accessible to users.

Definition at line 460 of file basic_ios.h.

Referenced by `std::basic_ios< char, _Traits >::basic_ios()`.

5.613.5 Member Function Documentation

5.613.5.1 `const locale& std::ios_base::M_getloc () const` `[inline], [inherited]`

Locale access.

Returns

A reference to the current locale.

Like `getloc` above, but returns a reference instead of generating a copy.

Definition at line 774 of file ios_base.h.

Referenced by `std::time_get< _CharT, _Inlter >::do_date_order()`, `std::time_get< _CharT, _Inlter >::do_get()`, `std::money_get< _CharT, _Inlter >::do_get()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::time_get< _CharT, _Inlter >::do_get_date()`, `std::time_get< _CharT, _Inlter >::do_get_monthname()`, `std::time_get< _CharT, _Inlter >::do_get_time()`, `std::time_get< _CharT, _Inlter >::do_get_weekday()`, `std::time_get< _CharT, _Inlter >::do_get_year()`, `std::time_put< _CharT, _Outlter >::do_put()`, `std::num_put< _CharT, _Outlter >::do_put()`, `std::time_get< _CharT, _Inlter >::get()`, and `std::time_put< _CharT, _Outlter >::put()`.

5.613.5.2 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::bad () const`
`[inline]`

Fast error checking.

Returns

True if the badbit is set.

Note that other iostate flags may also be set.

Definition at line 211 of file basic_ios.h.

5.613.5.3 `template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits >::clear (iostate __state =`
`goodbit)`

[Re]sets the error state.

Parameters

<code>__state</code>	The new state flag(s) to set.
----------------------	-------------------------------

See `std::ios_base::iostate` for the possible bit values. Most users will not need to pass an argument.

Definition at line 41 of file `basic_ios.tcc`.

Referenced by `std::basic_ios< char, _Traits >::exceptions()`, `std::__detail::operator>>()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_ios< char, _Traits >::rdstate()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ios< char, _Traits >::setstate()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

5.613.5.4 `template<typename _CharT, typename _Traits> basic_ios< _CharT, _Traits> & std::basic_ios< _CharT, _Traits>::copyfmt (const basic_ios< _CharT, _Traits> & __rhs)`

Copies fields of `__rhs` into this.

Parameters

<code>__rhs</code>	The source values for the copies.
--------------------	-----------------------------------

Returns

Reference to this object.

All fields of `__rhs` are copied into this object except that `rdbuf()` and `rdstate()` remain unchanged. All values in the `pword` and `iword` arrays are copied. Before copying, each callback is invoked with `erase_event`. After copying, each (new) callback is invoked with `copyfmt_event`. The final step is to copy `exceptions()`.

Definition at line 63 of file `basic_ios.tcc`.

References `std::basic_ios< _CharT, _Traits>::exceptions()`, `std::basic_ios< _CharT, _Traits>::fill()`, `std::ios_base::flags()`, `std::ios_base::getloc()`, `std::ios_base::precision()`, `std::basic_ios< _CharT, _Traits>::tie()`, `std::tie()`, and `std::ios_base::width()`.

Referenced by `std::basic_ios< char, _Traits>::rdbuf()`.

5.613.5.5 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits>::eof () const [inline]`

Fast error checking.

Returns

True if the eofbit is set.

Note that other `iostate` flags may also be set.

Definition at line 190 of file `basic_ios.h`.

5.613.5.6 `template<typename _CharT, typename _Traits> iostate std::basic_ios<_CharT, _Traits>::exceptions () const`
`[inline]`

Throwing exceptions on errors.

Returns

The current exceptions mask.

This changes nothing in the stream. See the one-argument version of `exceptions(iostate)` for the meaning of the return value.

Definition at line 222 of file `basic_ios.h`.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, and `std::basic_ios<char, _Traits>::setstate()`.

5.613.5.7 `template<typename _CharT, typename _Traits> void std::basic_ios<_CharT, _Traits>::exceptions (iostate`
`__except) [inline]`

Throwing exceptions on errors.

Parameters

<code>__except</code>	The new exceptions mask.
-----------------------	--------------------------

By default, error flags are set silently. You can set an exceptions mask for each stream; if a bit in the mask becomes set in the error flags, then an exception of type `std::ios_base::failure` is thrown.

If the error flag is already set when the exceptions mask is added, the exception is immediately thrown. Try running the following under GCC 3.1 or later:

```
1 #include <iostream>
2 #include <fstream>
3 #include <exception>
4
5 int main()
6 {
7     std::set_terminate (__gnu_cxx::__verbose_terminate_handler);
8
9     std::ifstream f ("/etc/motd");
10
11     std::cerr << "Setting badbit\n";
12     f.setstate (std::ios_base::badbit);
13
14     std::cerr << "Setting exception mask\n";
15     f.exceptions (std::ios_base::badbit);
16 }
```

Definition at line 257 of file `basic_ios.h`.

5.613.5.8 `template<typename _CharT, typename _Traits> bool std::basic_ios<_CharT, _Traits>::fail () const` `[inline]`

Fast error checking.

Returns

True if either the badbit or the failbit is set.

Checking the badbit in fail() is historical practice. Note that other iostate flags may also be set.

Definition at line 201 of file basic_ios.h.

Referenced by std::basic_ios< char, _Traits >::operator bool(), std::basic_ios< char, _Traits >::operator!(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_istream< _CharT, _Traits >::tellg(), std::basic_ostream< _CharT, _Traits >::tellp(), and std::regex_traits< _Ch_type >::value().

5.613.5.9 `template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::fill () const`
`[inline]`

Retrieves the *empty* character.

Returns

The current fill character.

It defaults to a space (' ') in the current locale.

Definition at line 370 of file basic_ios.h.

Referenced by std::basic_ios< _CharT, _Traits >::copyfmt(), std::basic_ios< char, _Traits >::fill(), std::normal_distribution< _RealType >::operator()(), std::gamma_distribution< _RealType >::operator()(), std::negative_binomial_distribution< _IntType >::operator()(), and std::operator>>().

5.613.5.10 `template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::fill (char_type`
`__ch) [inline]`

Sets a new *empty* character.

Parameters

<code>__ch</code>	The new character.
-------------------	--------------------

Returns

The previous fill character.

The fill character is used to fill out space when P+ characters have been requested (e.g., via setw), Q characters are actually used, and Q<P. It defaults to a space (' ') in the current locale.

Definition at line 390 of file basic_ios.h.

5.613.5.11 `fmtflags std::ios_base::flags () const` `[inline],[inherited]`

Access to format flags.

Returns

The format control flags for both input and output.

Definition at line 619 of file `ios_base.h`.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::mersenne_twister_engine<_UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f>::discard()`, `std::money_get<_CharT, _InIter>::do_get()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_put<_CharT, _OutIter>::do_put()`, `std::subtract_with_carry_engine<_UIntType, __w, __s, __r>::operator()()`, `std::discard_block_engine<_RandomNumberEngine, __p, __r>::operator()()`, `std::shuffle_order_engine<_RandomNumberEngine, __k>::operator()()`, `std::normal_distribution<_RealType>::operator()()`, `std::gamma_distribution<_RealType>::operator()()`, `std::binomial_distribution<_IntType>::operator()()`, `std::negative_binomial_distribution<_IntType>::operator()()`, `std::poisson_distribution<_IntType>::operator()()`, `std::basic_ostream<_CharT, _Traits>::operator<<()`, `std::operator<<()`, `std::__detail::operator>>()`, `std::operator>>()`, `std::linear_congruential_engine<_UIntType, __a, __c, __m>::seed()`, and `std::basic_istream<_CharT, _Traits>::sentry::sentry()`.

5.613.5.12 `fmtflags std::ios_base::flags (fmtflags __fmtfl)` `[inline],[inherited]`

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

Definition at line 630 of file `ios_base.h`.

5.613.5.13 `locale std::ios_base::getloc () const` `[inline],[inherited]`

Locale access.

Returns

A copy of the current locale.

If `imbue(loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale()`, the global C++ locale.

Definition at line 763 of file `ios_base.h`.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::money_put<_CharT, _OutIter>::do_put()`, `std::operator>>()`, and `std::ws()`.

5.613.5.14 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::good () const`
`[inline]`

Fast error checking.

Returns

True if no error flags are set.

A wrapper around `rdstate`.

Definition at line 180 of file `basic_ios.h`.

Referenced by `std::__detail::operator>>()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_istream< _CharT, _Traits >::sentry::sentry()`.

5.613.5.15 `template<typename _CharT, typename _Traits> locale std::basic_ios< _CharT, _Traits >::imbue (const locale & __loc)`

Moves to a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls `ios_base::imbue(loc)`, and if a stream buffer is associated with this stream, calls that buffer's `pubimbue(loc)`.

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>.

Definition at line 114 of file `basic_ios.tcc`.

References `std::ios_base::imbue()`.

Referenced by `std::basic_ios< char, _Traits >::fill()`.

5.613.5.16 `template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits >::init (basic_streambuf< _CharT, _Traits > * __sb)` `[protected]`

All setup is performed here.

This is called from the public constructor. It is not virtual and cannot be redefined.

Definition at line 126 of file `basic_ios.tcc`.

Referenced by `std::basic_ios< char, _Traits >::basic_ios()`.

5.613.5.17 `long& std::ios_base::iword (int __ix)` `[inline]`, `[inherited]`

Access to integer array.

Parameters

<code>_↔ _ix</code>	Index into the array.
-------------------------	-----------------------

Returns

A reference to an integer associated with the index.

The `iword` function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 809 of file `ios_base.h`.

5.613.5.18 `template<typename _CharT, typename _Traits> char std::basic_ios<_CharT, _Traits>::narrow (char_type __c, char __default) const [inline]`

Squeezes characters.

Parameters

<code>__c</code>	The character to narrow.
<code>__default</code>	The character to narrow.

Returns

The narrowed character.

Maps a character of `char_type` to a character of `char`, if possible.

Returns the result of

```
1 std::use_facet<ctype<char_type>> >(getloc()).narrow(c, default)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Definition at line 430 of file `basic_ios.h`.

5.613.5.19 `template<typename _CharT, typename _Traits> std::basic_ios<_CharT, _Traits>::operator bool () const [inline], [explicit]`

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`

Definition at line 117 of file `basic_ios.h`.

5.613.5.20 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::operator! () const`
`[inline]`

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`.

Definition at line 125 of file `basic_ios.h`.

5.613.5.21 `streamsize std::ios_base::precision () const` `[inline]`, `[inherited]`

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Definition at line 689 of file `ios_base.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::normal_distribution< _RealType >::operator()()`, `std::gamma_distribution< _RealType >::operator()()`, `std::negative_binomial_distribution< _IntType >::operator()()`, and `std::operator>>()`.

5.613.5.22 `streamsize std::ios_base::precision (streamsize __prec)` `[inline]`, `[inherited]`

Changing flags.

Parameters

<code>__prec</code>	The new precision value.
---------------------	--------------------------

Returns

The previous value of `precision()`.

Definition at line 698 of file `ios_base.h`.

5.613.5.23 `void*& std::ios_base::pword (int __ix)` `[inline]`, `[inherited]`

Access to void pointer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to a `void*` associated with the index.

The `pwd` function provides access to an array of pointers that can be used for any purpose. The array grows as required to hold the supplied index. All pointers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 830 of file `ios_base.h`.

```
5.613.5.24  template<typename _CharT, typename _Traits> basic_streambuf<_CharT, _Traits>* std::basic_ios<_CharT,
           _Traits>::rdbuf( ) const    [inline]
```

Accessing the underlying buffer.

Returns

The current stream buffer.

This does not change the state of the stream.

Definition at line 321 of file `basic_ios.h`.

Referenced by `std::basic_ostream<_CharT, _Traits>::flush()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_ostream<_CharT, _Traits>::operator<<()`, `std::tr2::operator>>()`, `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::operator>>()`, `std::basic_istream<_CharT, _Traits>::peek()`, `std::basic_ostream<_CharT, _Traits>::put()`, `std::basic_istream<_CharT, _Traits>::putback()`, `std::basic_ios<char, _Traits>::rdbuf()`, `std::basic_istream<_CharT, _Traits>::read()`, `std::basic_istream<_CharT, _Traits>::readsome()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_ostream<_CharT, _Traits>::seekp()`, `std::basic_istream<_CharT, _Traits>::sentry()`, `std::sentry()`, `std::basic_istream<_CharT, _Traits>::sync()`, `std::basic_istream<_CharT, _Traits>::tellg()`, `std::basic_ostream<_CharT, _Traits>::tellp()`, `std::basic_istream<_CharT, _Traits>::unget()`, and `std::ws()`.

```
5.613.5.25  template<typename _CharT, typename _Traits> basic_streambuf<_CharT, _Traits> * std::basic_ios<_CharT,
           _Traits>::rdbuf( basic_streambuf<_CharT, _Traits> * __sb )
```

Changing the underlying buffer.

Parameters

<code>__sb</code>	The new stream buffer.
-------------------	------------------------

Returns

The previous stream buffer.

Associates a new buffer with the current stream, and clears the error state.

Due to historical accidents which the LWG refuses to correct, the I/O library suffers from a design error: this function is hidden in derived classes by overrides of the zero-argument `rdbuf()`, which is non-virtual for hysterical raisins. As a result, you must use explicit qualifications to access this function via any derived class. For example:

```
1 std::fstream    foo;           // or some other derived type
2 std::streambuf* p = .....;
3
4 foo.ios::rdbuf(p);           // ios == basic_ios<char>
```

Definition at line 53 of file `basic_ios.tcc`.

5.613.5.26 `template<typename _CharT, typename _Traits> istate std::basic_ios< _CharT, _Traits >::rdstate () const`
`[inline]`

Returns the error state of the stream buffer.

Returns

A bit pattern (well, isn't everything?)

See `std::ios_base::istate` for the possible bit values. Most users will call one of the interpreting wrappers, e.g., `good()`.

Definition at line 137 of file `basic_ios.h`.

Referenced by `std::basic_ios< char, _Traits >::bad()`, `std::basic_ios< char, _Traits >::eof()`, `std::basic_ios< char, _Traits >::fail()`, `std::basic_ios< char, _Traits >::good()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ios< char, _Traits >::setstate()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

5.613.5.27 `void std::ios_base::register_callback (event_callback __fn, int __index)` `[inherited]`

Add the callback `__fn` with parameter `__index`.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

5.613.5.28 `fmtflags std::ios_base::setf (fmtflags __fmtfl)` `[inline],[inherited]`

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
----------------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set.

Definition at line 646 of file `ios_base.h`.

Referenced by `std::boolalpha()`, `std::dec()`, `std::fixed()`, `std::hex()`, `std::hexfloat()`, `std::left()`, `std::oct()`, `std::__detail<::operator>>()`, `std::right()`, `std::scientific()`, `std::showbase()`, `std::showpoint()`, `std::showpos()`, `std::skipws()`, `std::unitbuf()`, and `std::uppercase()`.

5.613.5.29 `fmtflags std::ios_base::setf (fmtflags __fmtfl, fmtflags __mask)` `[inline]`, `[inherited]`

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
<code>__mask</code>	The flags mask for <code>fmtfl</code> .

Returns

The previous format control flags.

This function clears `mask` in the format flags, then sets `fmtfl` & `mask`. An example mask is `ios_base::adjustfield`.

Definition at line 663 of file `ios_base.h`.

5.613.5.30 `template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits >::setstate (iostate __state)` `[inline]`

Sets additional flags in the error state.

Parameters

<code>__state</code>	The additional state flag(s) to set.
----------------------	--------------------------------------

See `std::ios_base::iostate` for the possible bit values.

Definition at line 157 of file `basic_ios.h`.

Referenced by `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::tr2::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT,`

_Traits >::read(), std::basic_istream< _CharT, _Traits >::readsome(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_ostream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::sync(), std::basic_istream< _CharT, _Traits >::unget(), and std::ws().

5.613.5.31 static bool std::ios_base::sync_with_stdio (bool __sync = true) [static], [inherited]

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., stdout) and the standard C++ objects (e.g., cout). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>

5.613.5.32 template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>* std::basic_ios< _CharT, _Traits >::tie () const [inline]

Fetches the current *tied* stream.

Returns

A pointer to the tied stream, or NULL if the stream is not tied.

A stream may be *tied* (or synchronized) to a second output stream. When this stream performs any I/O, the tied stream is first flushed. For example, `std::cin` is tied to `std::cout`.

Definition at line 295 of file basic_ios.h.

Referenced by std::basic_ios< char, _Traits >::basic_ios(), std::basic_ios< _CharT, _Traits >::copyfmt(), std::basic_ostream< _CharT, _Traits >::sentry::sentry(), and std::basic_istream< _CharT, _Traits >::sentry::sentry().

5.613.5.33 template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>* std::basic_ios< _CharT, _Traits >::tie (basic_ostream< _CharT, _Traits > * __tiestr) [inline]

Ties this stream to an output stream.

Parameters

<code>__tiestr</code>	The output stream.
-----------------------	--------------------

Returns

The previously tied output stream, or NULL if the stream was not tied.

This sets up a new tie; see `tie()` for more.

Definition at line 307 of file `basic_ios.h`.

5.613.5.34 `void std::ios_base::unsetf (fmtflags __mask) [inline],[inherited]`

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

Definition at line 678 of file `ios_base.h`.

Referenced by `std::defaultfloat()`, `std::noboolalpha()`, `std::noshowbase()`, `std::noshowpoint()`, `std::noshowpos()`, `std::noskipws()`, `std::nounitbuf()`, and `std::nouppercase()`.

5.613.5.35 `template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::widen (char __c) const [inline]`

Widens characters.

Parameters

<code>__c</code>	The character to widen.
------------------	-------------------------

Returns

The widened character.

Maps a character of `char` to a character of `char_type`.

Returns the result of

```
1 std::use_facet<ctype<char_type> > (getloc()).widen(c)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Definition at line 449 of file `basic_ios.h`.

Referenced by `std::basic_ios< char, _Traits >::fill()`, `std::getline()`, `std::normal_distribution< _RealType >::operator()()`, `std::gamma_distribution< _RealType >::operator()()`, `std::negative_binomial_distribution< _IntType >::operator()()`, `std::tr2::operator>>()`, and `std::operator>>()`.

5.613.5.36 `streamsize std::ios_base::width () const` `[inline],[inherited]`

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Definition at line 712 of file ios_base.h.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::money_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, and `std::operator>>()`.

5.613.5.37 `streamsize std::ios_base::width (streamsize __wide)` `[inline],[inherited]`

Changing flags.

Parameters

<code>__wide</code>	The new width value.
---------------------	----------------------

Returns

The previous value of `width()`.

Definition at line 721 of file ios_base.h.

5.613.5.38 `static int std::ios_base::xalloc () throw` `[static],[inherited]`

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the `iword` and `pword` functions. The expectation is that an application calls `xalloc` in order to obtain an index in the `iword` and `pword` arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. `xalloc` is guaranteed to return an index that is safe to use in the `iword` and `pword` arrays.

5.613.6 Member Data Documentation

5.613.6.1 `const fmtflags std::ios_base::adjustfield` `[static], [inherited]`

A mask of left|right|internal. Useful for the 2-arg form of `setf`.

Definition at line 378 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::internal()`, `std::left()`, and `std::right()`.

5.613.6.2 `const openmode std::ios_base::app` `[static], [inherited]`

Seek to end before each write.

Definition at line 432 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

5.613.6.3 `const openmode std::ios_base::ate` `[static], [inherited]`

Open and seek to end immediately after opening.

Definition at line 435 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::open()`.

5.613.6.4 `const iostate std::ios_base::badbit` `[static], [inherited]`

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Definition at line 402 of file `ios_base.h`.

Referenced by `std::basic_ios< char, _Traits >::bad()`, `std::basic_ios< char, _Traits >::fail()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::basic_ostream< _CharT, _Traits >::write()`.

5.613.6.5 `const fmtflags std::ios_base::basefield` `[static], [inherited]`

A mask of dec|oct|hex. Useful for the 2-arg form of `setf`.

Definition at line 381 of file `ios_base.h`.

Referenced by `std::dec()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::hex()`, `std::oct()`, and `std::basic_ostream< _CharT, _Traits >::operator<<()`.

5.613.6.6 const seekdir std::ios_base::beg [static],[inherited]

Request a seek relative to the beginning of the stream.

Definition at line 464 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::seekpos(), and __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync().

5.613.6.7 const openmode std::ios_base::binary [static],[inherited]

Perform input and output in binary mode (as opposed to text mode). This is probably not what you think it is; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>.

Definition at line 440 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::showmanyc().

5.613.6.8 const fmtflags std::ios_base::boolalpha [static],[inherited]

Insert/extract `bool` in alphabetic rather than numeric format.

Definition at line 326 of file ios_base.h.

Referenced by std::boolalpha(), std::num_get< _CharT, _InIter >::do_get(), std::num_put< _CharT, _OutIter >::do_put(), and std::noboolalpha().

5.613.6.9 const seekdir std::ios_base::cur [static],[inherited]

Request a seek relative to the current position within the sequence.

Definition at line 467 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::imbue(), std::basic_filebuf< _CharT, _Traits >::overflow(), std::basic_filebuf< _CharT, _Traits >::pbackfail(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff(), std::basic_filebuf< _CharT, _Traits >::seekoff(), __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync(), std::basic_istream< _CharT, _Traits >::tellg(), and std::basic_ostream< _CharT, _Traits >::tellp().

5.613.6.10 const fmtflags std::ios_base::dec [static],[inherited]

Converts integer input or generates integer output in decimal base.

Definition at line 329 of file ios_base.h.

Referenced by std::dec().

5.613.6.11 `const seekdir std::ios_base::end` `[static], [inherited]`

Request a seek relative to the current end of the sequence.

Definition at line 470 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::open()`, and `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`.

5.613.6.12 `const iostate std::ios_base::eofbit` `[static], [inherited]`

Indicates that an input operation reached the end of an input sequence.

Definition at line 405 of file `ios_base.h`.

Referenced by `std::time_get< _CharT, _Inlter >::do_get()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::time_get< _CharT, _Inlter >::do_get_date()`, `std::time_get< _CharT, _Inlter >::do_get_monthname()`, `std::time_get< _CharT, _Inlter >::do_get_time()`, `std::time_get< _CharT, _Inlter >::do_get_weekday()`, `std::time_get< _CharT, _Inlter >::do_get_year()`, `std::basic_ios< char, _Traits >::eof()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::time_get< _CharT, _Inlter >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::ws()`.

5.613.6.13 `const iostate std::ios_base::failbit` `[static], [inherited]`

Indicates that an input operation failed to read the expected characters, or that an output operation failed to generate the desired characters.

Definition at line 410 of file `ios_base.h`.

Referenced by `std::time_get< _CharT, _Inlter >::do_date_order()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::time_get< _CharT, _Inlter >::do_get_monthname()`, `std::time_get< _CharT, _Inlter >::do_get_weekday()`, `std::time_get< _CharT, _Inlter >::do_get_year()`, `std::basic_ios< char, _Traits >::fail()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::time_get< _CharT, _Inlter >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_istream< _CharT, _Traits >::sentry::sentry()`.

5.613.6.14 `const fmtflags std::ios_base::fixed` `[static], [inherited]`

Generate floating-point output in fixed-point notation.

Definition at line 332 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _Inlter >::do_get()`, `std::fixed()`, and `std::hexfloat()`.

5.613.6.15 `const fmtflags std::ios_base::floatfield` `[static], [inherited]`

A mask of scientific|fixed. Useful for the 2-arg form of `setf`.

Definition at line 384 of file `ios_base.h`.

Referenced by `std::defaultfloat()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::fixed()`, `std::hexfloat()`, and `std::scientific()`.

5.613.6.16 `const iostate std::ios_base::goodbit` `[static], [inherited]`

Indicates all is well.

Definition at line 413 of file `ios_base.h`.

Referenced by `std::time_get< _CharT, _Inlter >::do_date_order()`, `std::time_get< _CharT, _Inlter >::do_get()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::time_get< _CharT, _Inlter >::do_get_monthname()`, `std::time_get< _CharT, _Inlter >::do_get_weekday()`, `std::time_get< _CharT, _Inlter >::do_get_year()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::time_get< _CharT, _Inlter >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_ios< char, _Traits >::rdstate()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sync()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

5.613.6.17 `const fmtflags std::ios_base::hex` `[static], [inherited]`

Converts integer input or generates integer output in hexadecimal base.

Definition at line 335 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _Inlter >::do_get()`, `std::num_put< _CharT, _Outlter >::do_put()`, `std::hex()`, and `std::basic_ostream< _CharT, _Traits >::operator<<()`.

5.613.6.18 `const openmode std::ios_base::in` `[static], [inherited]`

Open for input. Default for `ifstream` and `fstream`.

Definition at line 443 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos()`, `std::basic_filebuf< _CharT, _Traits >::showmanyc()`, `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::underflow()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

5.613.6.19 `const fmtflags std::ios_base::internal` `[static], [inherited]`

Adds fill characters at a designated internal point in certain generated output, or identical to `right` if no such point is designated.

Definition at line 340 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, and `std::ios_base::internal()`.

5.613.6.20 `const fmtflags std::ios_base::left` `[static], [inherited]`

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.)

Definition at line 344 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, and `std::ios_base::left()`.

5.613.6.21 `const fmtflags std::ios_base::oct` `[static], [inherited]`

Converts integer input or generates integer output in octal base.

Definition at line 347 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _InIter >::do_get()`, `std::oct()`, and `std::basic_ostream< _CharT, _Traits >::operator<<()`.

5.613.6.22 `const openmode std::ios_base::out` `[static], [inherited]`

Open for output. Default for `ofstream` and `fstream`.

Definition at line 446 of file `ios_base.h`.

Referenced by `std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow()`, `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::pbackfail()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos()`, `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, and `std::basic_filebuf< _CharT, _Traits >::xspn()`.

5.613.6.23 `const fmtflags std::ios_base::right` `[static], [inherited]`

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Definition at line 351 of file `ios_base.h`.

Referenced by `std::right()`.

5.613.6.24 `const fmtflags std::ios_base::scientific` `[static], [inherited]`

Generates floating-point output in scientific notation.

Definition at line 354 of file `ios_base.h`.

Referenced by `std::hexfloat()`, and `std::scientific()`.

5.613.6.25 `const fmtflags std::ios_base::showbase` `[static], [inherited]`

Generates a prefix indicating the numeric base of generated integer output.

Definition at line 358 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _OutIter >::do_put()`, `std::noshowbase()`, and `std::showbase()`.

5.613.6.26 `const fmtflags std::ios_base::showpoint` `[static], [inherited]`

Generates a decimal-point character unconditionally in generated floating-point output.

Definition at line 362 of file `ios_base.h`.

Referenced by `std::noshowpoint()`, and `std::showpoint()`.

5.613.6.27 `const fmtflags std::ios_base::showpos` `[static], [inherited]`

Generates a + sign in non-negative generated numeric output.

Definition at line 365 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _InIter >::do_get()`, `std::noshowpos()`, and `std::showpos()`.

5.613.6.28 `const fmtflags std::ios_base::skipws` `[static], [inherited]`

Skips leading white space before certain input operations.

Definition at line 368 of file `ios_base.h`.

Referenced by `std::noskipws()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, and `std::skipws()`.

5.613.6.29 `const openmode std::ios_base::trunc` `[static], [inherited]`

Open for input. Default for `ofstream`.

Definition at line 449 of file `ios_base.h`.

5.613.6.30 `const fmtflags std::ios_base::unitbuf` `[static], [inherited]`

Flushes output after each output operation.

Definition at line 371 of file `ios_base.h`.

Referenced by `std::nounitbuf()`, and `std::unitbuf()`.

5.613.6.31 `const fmtflags std::ios_base::uppercase` `[static], [inherited]`

Replaces certain lowercase letters with their uppercase equivalents in generated output.

Definition at line 375 of file `ios_base.h`.

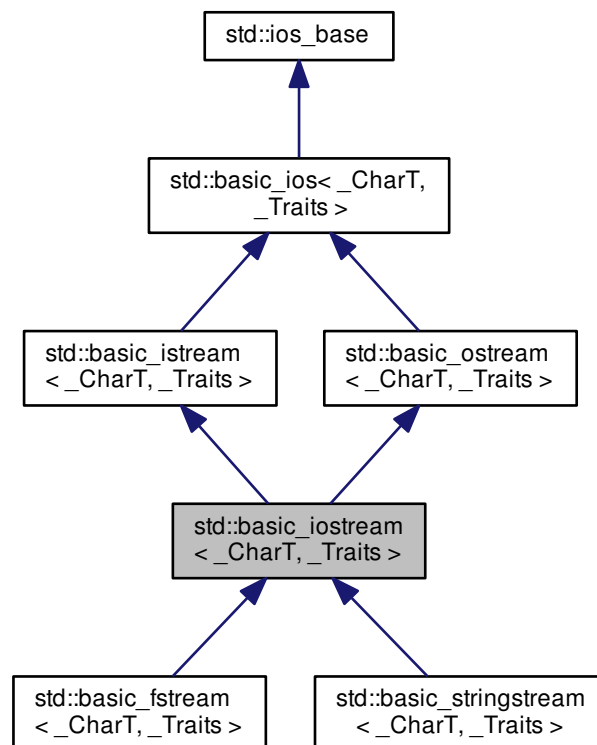
Referenced by `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::nouppercase()`, and `std::uppercase()`.

The documentation for this class was generated from the following files:

- [iosfwd](#)
- [basic_ios.h](#)
- [basic_ios.tcc](#)

5.614 `std::basic_iostream< _CharT, _Traits >` Class Template Reference

Inheritance diagram for `std::basic_iostream< _CharT, _Traits >`:



Public Types

- typedef [ctype](#)< _CharT > **__ctype_type**
 - typedef [ctype](#)< _CharT > **__ctype_type**
 - typedef [basic_ios](#)< _CharT, _Traits > **__ios_type**
 - typedef [basic_ios](#)< _CharT, _Traits > **__ios_type**
 - typedef [basic_istream](#)< _CharT, _Traits > **__istream_type**
 - typedef [num_get](#)< _CharT, [istreambuf_iterator](#)< _CharT, _Traits > > **__num_get_type**
 - typedef [num_put](#)< _CharT, [ostreambuf_iterator](#)< _CharT, _Traits > > **__num_put_type**
 - typedef [basic_ostream](#)< _CharT, _Traits > **__ostream_type**
 - typedef [basic_streambuf](#)< _CharT, _Traits > **__streambuf_type**
 - typedef [basic_streambuf](#)< _CharT, _Traits > **__streambuf_type**
 - typedef _CharT **char_type**
 - enum [event](#) { [erase_event](#), [imbue_event](#), [copyfmt_event](#) }
 - typedef void(* [event_callback](#)) ([event](#) __e, [ios_base](#) & __b, int __i)
 - typedef _ios_Fmtflags **fmtflags**
 - typedef _Traits::int_type **int_type**
 - typedef int **io_state**
 - typedef _ios_iostate **iostate**
 - typedef _Traits::off_type **off_type**
 - typedef int **open_mode**
 - typedef _ios_Openmode **openmode**
 - typedef _Traits::pos_type **pos_type**
 - typedef int **seek_dir**
 - typedef _ios_Seekdir **seekdir**
 - typedef [std::streamoff](#) **streamoff**
 - typedef [std::streampos](#) **streampos**
 - typedef _Traits **traits_type**
-
- typedef [num_put](#)< _CharT, [ostreambuf_iterator](#)< _CharT, _Traits > > **__num_put_type**

Public Member Functions

- [basic_istream](#) ([basic_streambuf](#)< _CharT, _Traits > *__sb)
- virtual [~basic_istream](#) ()
- template<typename _ValueT >
[basic_istream](#)< _CharT, _Traits > & **_M_extract** (_ValueT & __v)
- const [locale](#) & **_M_getloc** () const
- template<typename _ValueT >
[basic_ostream](#)< _CharT, _Traits > & **_M_insert** (_ValueT __v)
- void **_M_setstate** ([iostate](#) __state)
- bool **bad** () const
- void **clear** ([iostate](#) __state=[goodbit](#))
- [basic_ios](#) & [copyfmt](#) (const [basic_ios](#) & __rhs)
- bool **eof** () const
- [iostate](#) **exceptions** () const
- void **exceptions** ([iostate](#) __except)
- bool **fail** () const
- char_type **fill** () const
- char_type **fill** (char_type __ch)

- `fmtflags flags () const`
 - `fmtflags flags (fmtflags __fmtfl)`
 - `__ostream_type & flush ()`
 - `streamsize gcount () const`
 - `template<>`
`basic_istream< char > & getline (char_type *__s, streamsize __n, char_type __delim)`
 - `template<>`
`basic_istream< wchar_t > & getline (char_type *__s, streamsize __n, char_type __delim)`
 - `locale getloc () const`
 - `bool good () const`
 - `template<>`
`basic_istream< char > & ignore (streamsize __n)`
 - `template<>`
`basic_istream< char > & ignore (streamsize __n, int_type __delim)`
 - `template<>`
`basic_istream< wchar_t > & ignore (streamsize __n)`
 - `template<>`
`basic_istream< wchar_t > & ignore (streamsize __n, int_type __delim)`
 - `locale imbue (const locale & __loc)`
 - `long & iword (int __ix)`
 - `char narrow (char_type __c, char __dfault) const`
 - `__ostream_type & operator<< (const void *__p)`
 - `__ostream_type & operator<< (__streambuf_type *__sb)`
 - `__istream_type & operator>> (void *& __p)`
 - `__istream_type & operator>> (__streambuf_type *__sb)`
 - `streamsize precision () const`
 - `streamsize precision (streamsize __prec)`
 - `void *& pword (int __ix)`
 - `basic_streambuf< _CharT, _Traits > * rdbuf () const`
 - `basic_streambuf< _CharT, _Traits > * rdbuf (basic_streambuf< _CharT, _Traits > *__sb)`
 - `iosstate rdstate () const`
 - `void register_callback (event_callback __fn, int __index)`
 - `__ostream_type & seekp (pos_type)`
 - `__ostream_type & seekp (off_type, ios_base::seekdir)`
 - `fmtflags setf (fmtflags __fmtfl)`
 - `fmtflags setf (fmtflags __fmtfl, fmtflags __mask)`
 - `void setstate (iosstate __state)`
 - `pos_type tellp ()`
 - `basic_ostream< _CharT, _Traits > * tie () const`
 - `basic_ostream< _CharT, _Traits > * tie (basic_ostream< _CharT, _Traits > *__tiestr)`
 - `void unsetf (fmtflags __mask)`
 - `char_type widen (char __c) const`
 - `streamsize width () const`
 - `streamsize width (streamsize __wide)`
-
- `__istream_type & operator>> (__istream_type &(*__pf)(__istream_type &))`
 - `__istream_type & operator>> (__ios_type &(*__pf)(__ios_type &))`
 - `__istream_type & operator>> (ios_base &(*__pf)(ios_base &))`

Extractors

All the `operator>>` functions (aka formatted input functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (`noskipws`) set to `false`. This has several effects, concluding with the setting of a status flag; see the `sentry` documentation for more.

If the `sentry` status is good, the function tries to extract whatever data is appropriate for the type of the argument.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- `__istream_type & operator>> (bool &__n)`
- `__istream_type & operator>> (short &__n)`
- `__istream_type & operator>> (unsigned short &__n)`
- `__istream_type & operator>> (int &__n)`
- `__istream_type & operator>> (unsigned int &__n)`
- `__istream_type & operator>> (long &__n)`
- `__istream_type & operator>> (unsigned long &__n)`
- `__istream_type & operator>> (long long &__n)`
- `__istream_type & operator>> (unsigned long long &__n)`
- `__istream_type & operator>> (float &__f)`
- `__istream_type & operator>> (double &__f)`
- `__istream_type & operator>> (long double &__f)`

Unformatted Input Functions

All the unformatted input functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (`noskipws`) set to `true`. This has several effects, concluding with the setting of a status flag; see the `sentry` documentation for more.

If the `sentry` status is good, the function tries to extract whatever data is appropriate for the type of the argument.

The number of characters extracted is stored for later retrieval by `gcount()`.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- `int_type get ()`
- `__istream_type & get (char_type &__c)`
- `__istream_type & get (char_type * __s, streamsize __n, char_type __delim)`
- `__istream_type & get (char_type * __s, streamsize __n)`
- `__istream_type & get (__streambuf_type & __sb, char_type __delim)`
- `__istream_type & get (__streambuf_type & __sb)`
- `__istream_type & getline (char_type * __s, streamsize __n, char_type __delim)`
- `__istream_type & getline (char_type * __s, streamsize __n)`
- `__istream_type & ignore (streamsize __n, int_type __delim)`
- `__istream_type & ignore (streamsize __n)`
- `__istream_type & ignore ()`
- `int_type peek ()`
- `__istream_type & read (char_type * __s, streamsize __n)`
- `streamsize readsome (char_type * __s, streamsize __n)`
- `__istream_type & putback (char_type __c)`
- `__istream_type & unget ()`
- `int sync ()`
- `pos_type tellg ()`
- `__istream_type & seekg (pos_type)`
- `__istream_type & seekg (off_type, ios_base::seekdir)`
- `operator bool () const`

- bool `operator!()` const
- `__ostream_type & operator<< (__ostream_type &(__pf)(__ostream_type &))`
- `__ostream_type & operator<< (__ios_type &(__pf)(__ios_type &))`
- `__ostream_type & operator<< (ios_base &(__pf)(ios_base &))`

Inserters

All the `operator<<` functions (aka formatted output functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This can have several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- `__ostream_type & operator<< (long __n)`
- `__ostream_type & operator<< (unsigned long __n)`
- `__ostream_type & operator<< (bool __n)`
- `__ostream_type & operator<< (short __n)`
- `__ostream_type & operator<< (unsigned short __n)`
- `__ostream_type & operator<< (int __n)`
- `__ostream_type & operator<< (unsigned int __n)`
- `__ostream_type & operator<< (long long __n)`
- `__ostream_type & operator<< (unsigned long long __n)`
- `__ostream_type & operator<< (double __f)`
- `__ostream_type & operator<< (float __f)`
- `__ostream_type & operator<< (long double __f)`

Unformatted Output Functions

All the unformatted output functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This has several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state. If `badbit` is on in the stream's exceptions mask, the exception will be rethrown without completing its actions.

- `__ostream_type & put (char_type __c)`
- void `_M_write` (const char_type * __s, streamsize __n)
- `__ostream_type & write` (const char_type * __s, streamsize __n)

Static Public Member Functions

- static bool `sync_with_stdio` (bool __sync=true)
- static int `xalloc` () throw ()

Static Public Attributes

- static const [fmtflags](#) [adjustfield](#)
- static const [openmode](#) [app](#)
- static const [openmode](#) [ate](#)
- static const [iosstate](#) [badbit](#)
- static const [fmtflags](#) [basefield](#)
- static const [seekdir](#) [beg](#)
- static const [openmode](#) [binary](#)
- static const [fmtflags](#) [boolalpha](#)
- static const [seekdir](#) [cur](#)
- static const [fmtflags](#) [dec](#)
- static const [seekdir](#) [end](#)
- static const [iosstate](#) [eofbit](#)
- static const [iosstate](#) [failbit](#)
- static const [fmtflags](#) [fixed](#)
- static const [fmtflags](#) [floatfield](#)
- static const [iosstate](#) [goodbit](#)
- static const [fmtflags](#) [hex](#)
- static const [openmode](#) [in](#)
- static const [fmtflags](#) [internal](#)
- static const [fmtflags](#) [left](#)
- static const [fmtflags](#) [oct](#)
- static const [openmode](#) [out](#)
- static const [fmtflags](#) [right](#)
- static const [fmtflags](#) [scientific](#)
- static const [fmtflags](#) [showbase](#)
- static const [fmtflags](#) [showpoint](#)
- static const [fmtflags](#) [showpos](#)
- static const [fmtflags](#) [skipws](#)
- static const [openmode](#) [trunc](#)
- static const [fmtflags](#) [unitbuf](#)
- static const [fmtflags](#) [uppercase](#)

Protected Types

- enum { [_S_local_word_size](#) }

Protected Member Functions

- **basic_iostream** (const [basic_iostream](#) &)=delete
- **basic_iostream** ([basic_iostream](#) &&__rhs)
- void [_M_cache_locale](#) (const [locale](#) &__loc)
- void [_M_call_callbacks](#) ([event](#) __ev) throw ()
- void [_M_dispose_callbacks](#) (void) throw ()
- template<typename _ValueT >
 [_istream_type](#) & [_M_extract](#) (_ValueT &__v)
- [_Words](#) & [_M_grow_words](#) (int __index, bool __iword)
- void [_M_init](#) () throw ()

- `template<typename _ValueT >`
`__ostream_type & _M_insert (_ValueT __v)`
- `void _M_move (ios_base &) noexcept`
- `void _M_swap (ios_base & __rhs) noexcept`
- `void init (basic_streambuf< _CharT, _Traits > * __sb)`
- `void move (basic_ios & __rhs)`
- `void move (basic_ios && __rhs)`
- `basic_ostream & operator= (const basic_ostream &)=delete`
- `basic_ostream & operator= (basic_ostream && __rhs)`
- `void set_rdbuf (basic_streambuf< _CharT, _Traits > * __sb)`
- `void swap (basic_ostream & __rhs)`
- `void swap (basic_ios & __rhs) noexcept`
- `void swap (basic_istream & __rhs)`
- `void swap (basic_ostream & __rhs)`

Protected Attributes

- `_Callback_list * _M_callbacks`
- `const __ctype_type * _M_ctype`
- `iosstate _M_exception`
- `char_type _M_fill`
- `bool _M_fill_init`
- `fmtflags _M_flags`
- `streamsize _M_gcount`
- `locale _M_ios_locale`
- `_Words _M_local_word [_S_local_word_size]`
- `const __num_get_type * _M_num_get`
- `const __num_put_type * _M_num_put`
- `streamsize _M_precision`
- `basic_streambuf< _CharT, _Traits > * _M_streambuf`
- `iosstate _M_streambuf_state`
- `basic_ostream< _CharT, _Traits > * _M_tie`
- `streamsize _M_width`
- `_Words * _M_word`
- `int _M_word_size`
- `_Words _M_word_zero`

5.614.1 Detailed Description

```
template<typename _CharT, typename _Traits>
class std::basic_ostream< _CharT, _Traits >
```

Template class `basic_ostream`.

Template Parameters

<code>_CharT</code>	Type of character stream.
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .

This class multiply inherits from the input and output stream classes simply to provide a single interface.

Definition at line 89 of file iosfwd.

5.614.2 Member Typedef Documentation

5.614.2.1 `template<typename _CharT, typename _Traits> typedef num_put<_CharT, ostreambuf_iterator<_CharT, _Traits>
> std::basic_ios<_CharT, _Traits>::__num_put_type [inherited]`

These are non-standard types.

Definition at line 89 of file basic_ios.h.

5.614.2.2 `typedef void(* std::ios_base::event_callback)(event __e, ios_base &__b, int __i) [inherited]`

The type of an event callback function.

Parameters

<code>__e</code>	One of the members of the event enum.
<code>__b</code>	Reference to the ios_base object.
<code>__i</code>	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several ios_base and basic_ios functions, specifically imbue(), copyfmt(), and ~ios().

Definition at line 504 of file ios_base.h.

5.614.2.3 `typedef _Ios_Fmtflags std::ios_base::fmtflags [inherited]`

This is a bitmask type.

`_Ios_Fmtflags` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `fmtflags` are:

- `boolalpha`
- `dec`
- `fixed`
- `hex`
- `internal`
- `left`

- oct
- right
- scientific
- showbase
- showpoint
- showpos
- skipws
- unitbuf
- uppercase
- adjustfield
- basefield
- floatfield

Definition at line 323 of file ios_base.h.

5.614.2.4 `typedef _ios_istate std::ios_base::istate` [inherited]

This is a bitmask type.

`_Ios_Istate` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `istate` are:

- badbit
- eofbit
- failbit
- goodbit

Definition at line 398 of file ios_base.h.

5.614.2.5 `typedef _ios_Openmode std::ios_base::openmode` [inherited]

This is a bitmask type.

`_Ios_Openmode` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `openmode` are:

- app
- ate
- binary
- in
- out
- trunc

Definition at line 429 of file ios_base.h.

5.614.2.6 typedef _Ios_Seekdir std::ios_base::seekdir [inherited]

This is an enumerated type.

_Ios_Seekdir is implementation-defined. Defined values of type seekdir are:

- beg
- cur, equivalent to SEEK_CUR in the C standard library.
- end, equivalent to SEEK_END in the C standard library.

Definition at line 461 of file ios_base.h.

5.614.3 Member Enumeration Documentation

5.614.3.1 enum std::ios_base::event [inherited]

The set of events that may be passed to an event callback.

erase_event is used during ~ios() and copyfmt(). imbue_event is used during imbue(). copyfmt_event is used during copyfmt().

Definition at line 487 of file ios_base.h.

5.614.4 Constructor & Destructor Documentation

5.614.4.1 template<typename _CharT, typename _Traits> std::basic_iostream< _CharT, _Traits >::basic_iostream (basic_streambuf< _CharT, _Traits > * __sb) [inline], [explicit]

Constructor does nothing.

Both of the parent classes are initialized with the same streambuf pointer passed to this constructor.

Definition at line 849 of file istream.

5.614.4.2 template<typename _CharT, typename _Traits> virtual std::basic_iostream< _CharT, _Traits >::~~basic_iostream () [inline], [virtual]

Destructor does nothing.

Definition at line 856 of file istream.

5.614.5 Member Function Documentation

5.614.5.1 `const locale& std::ios_base::_M_getloc () const` `[inline]`, `[inherited]`

Locale access.

Returns

A reference to the current locale.

Like `getloc` above, but returns a reference instead of generating a copy.

Definition at line 774 of file `ios_base.h`.

Referenced by `std::time_get<_CharT, _Inlter>::do_date_order()`, `std::time_get<_CharT, _Inlter>::do_get()`, `std::money_get<_CharT, _Inlter>::do_get()`, `std::num_get<_CharT, _Inlter>::do_get()`, `std::time_get<_CharT, _Inlter>::do_get_date()`, `std::time_get<_CharT, _Inlter>::do_get_monthname()`, `std::time_get<_CharT, _Inlter>::do_get_time()`, `std::time_get<_CharT, _Inlter>::do_get_weekday()`, `std::time_get<_CharT, _Inlter>::do_get_year()`, `std::time_put<_CharT, _Outlter>::do_put()`, `std::num_put<_CharT, _Outlter>::do_put()`, `std::time_get<_CharT, _Inlter>::get()`, and `std::time_put<_CharT, _Outlter>::put()`.

5.614.5.2 `template<typename _CharT, typename _Traits> void std::basic_ostream<_CharT, _Traits>::_M_write (const char_type * __s, streamsize __n)` `[inline]`, `[inherited]`

Core write functionality, without sentry.

Parameters

<code>__s</code>	The array to insert.
<code>__n</code>	Maximum number of characters to insert.

Definition at line 311 of file `ostream`.

Referenced by `std::basic_ostream<_CharT, _Traits>::write()`.

5.614.5.3 `template<typename _CharT, typename _Traits> bool std::basic_ios<_CharT, _Traits>::bad () const` `[inline]`, `[inherited]`

Fast error checking.

Returns

True if the badbit is set.

Note that other `iostate` flags may also be set.

Definition at line 211 of file `basic_ios.h`.

5.614.5.4 `template<typename _CharT, typename _Traits> void std::basic_ios<_CharT, _Traits>::clear (iostate __state =
goodbit) [inherited]`

[Re]sets the error state.

Parameters

<code>__state</code>	The new state flag(s) to set.
----------------------	-------------------------------

See `std::ios_base::iostate` for the possible bit values. Most users will not need to pass an argument.

Definition at line 41 of file `basic_ios.tcc`.

Referenced by `std::basic_ios< char, _Traits >::exceptions()`, `std::__detail::operator>>()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_ios< char, _Traits >::rdstate()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ios< char, _Traits >::setstate()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

5.614.5.5 `template<typename _CharT, typename _Traits> basic_ios< _CharT, _Traits> & std::basic_ios< _CharT, _Traits>::copyfmt (const basic_ios< _CharT, _Traits> & __rhs) [inherited]`

Copies fields of `__rhs` into this.

Parameters

<code>__rhs</code>	The source values for the copies.
--------------------	-----------------------------------

Returns

Reference to this object.

All fields of `__rhs` are copied into this object except that `rdbuf()` and `rdstate()` remain unchanged. All values in the `pword` and `iword` arrays are copied. Before copying, each callback is invoked with `erase_event`. After copying, each (new) callback is invoked with `copyfmt_event`. The final step is to copy `exceptions()`.

Definition at line 63 of file `basic_ios.tcc`.

References `std::basic_ios< _CharT, _Traits>::exceptions()`, `std::basic_ios< _CharT, _Traits>::fill()`, `std::ios_base::flags()`, `std::ios_base::getloc()`, `std::ios_base::precision()`, `std::basic_ios< _CharT, _Traits>::tie()`, `std::tie()`, and `std::ios_base::width()`.

Referenced by `std::basic_ios< char, _Traits>::rdbuf()`.

5.614.5.6 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits>::eof () const [inline], [inherited]`

Fast error checking.

Returns

True if the eofbit is set.

Note that other `iostate` flags may also be set.

Definition at line 190 of file `basic_ios.h`.

5.614.5.7 `template<typename _CharT, typename _Traits> iostate std::basic_ios< _CharT, _Traits >::exceptions () const`
`[inline], [inherited]`

Throwing exceptions on errors.

Returns

The current exceptions mask.

This changes nothing in the stream. See the one-argument version of `exceptions(iostate)` for the meaning of the return value.

Definition at line 222 of file `basic_ios.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, and `std::basic_ios< char, _Traits >::setstate()`.

5.614.5.8 `template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits >::exceptions (iostate __except)`
`[inline], [inherited]`

Throwing exceptions on errors.

Parameters

<code>__except</code>	The new exceptions mask.
-----------------------	--------------------------

By default, error flags are set silently. You can set an exceptions mask for each stream; if a bit in the mask becomes set in the error flags, then an exception of type `std::ios_base::failure` is thrown.

If the error flag is already set when the exceptions mask is added, the exception is immediately thrown. Try running the following under GCC 3.1 or later:

```
1 #include <iostream>
2 #include <fstream>
3 #include <exception>
4
5 int main()
6 {
7     std::set_terminate (__gnu_cxx::__verbose_terminate_handler);
8
9     std::ifstream f ("/etc/motd");
10
11     std::cerr << "Setting badbit\n";
12     f.setstate (std::ios_base::badbit);
13
14     std::cerr << "Setting exception mask\n";
15     f.exceptions (std::ios_base::badbit);
16 }
```

Definition at line 257 of file `basic_ios.h`.

5.614.5.9 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::fail () const`
`[inline], [inherited]`

Fast error checking.

Returns

True if either the badbit or the failbit is set.

Checking the badbit in fail() is historical practice. Note that other iostate flags may also be set.

Definition at line 201 of file basic_ios.h.

Referenced by std::basic_ios< char, _Traits >::operator bool(), std::basic_ios< char, _Traits >::operator!(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_istream< _CharT, _Traits >::tellg(), std::basic_ostream< _CharT, _Traits >::tellp(), and std::regex_traits< _Ch_type >::value().

5.614.5.10 `template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::fill () const`
`[inline], [inherited]`

Retrieves the *empty* character.

Returns

The current fill character.

It defaults to a space (' ') in the current locale.

Definition at line 370 of file basic_ios.h.

Referenced by std::basic_ios< _CharT, _Traits >::copyfmt(), std::basic_ios< char, _Traits >::fill(), std::normal_distribution< _RealType >::operator()(), std::gamma_distribution< _RealType >::operator()(), std::negative_binomial_distribution< _IntType >::operator()(), and std::operator>>().

5.614.5.11 `template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::fill (char_type __ch`
`) [inline], [inherited]`

Sets a new *empty* character.

Parameters

<code>__ch</code>	The new character.
-------------------	--------------------

Returns

The previous fill character.

The fill character is used to fill out space when P+ characters have been requested (e.g., via setw), Q characters are actually used, and Q<P. It defaults to a space (' ') in the current locale.

Definition at line 390 of file basic_ios.h.

5.614.5.12 **fmtflags** std::ios_base::flags () const [inline],[inherited]

Access to format flags.

Returns

The format control flags for both input and output.

Definition at line 619 of file ios_base.h.

Referenced by std::basic_ios< _CharT, _Traits >::copyfmt(), std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >::discard(), std::money_get< _CharT, _InIter >::do_get(), std::num_get< _CharT, _InIter >::do_get(), std::num_put< _CharT, _OutIter >::do_put(), std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::operator>(), std::discard_block_engine< _RandomNumberEngine, __p, __r >::operator>(), std::shuffle_order_engine< _RandomNumberEngine, __k >::operator>(), std::normal_distribution< _RealType >::operator>(), std::gamma_distribution< _RealType >::operator>(), std::binomial_distribution< _IntType >::operator>(), std::negative_binomial_distribution< _IntType >::operator>(), std::poisson_distribution< _IntType >::operator(), std::basic_ostream< _CharT, _Traits >::operator<<(), std::operator<<(), std::__detail::operator>>(), std::operator>>(), std::linear_congruential_engine< _UIntType, __a, __c, __m >::seed(), and std::basic_istream< _CharT, _Traits >::sentry::sentry().

5.614.5.13 **fmtflags** std::ios_base::flags (fmtflags __fmtfl) [inline],[inherited]

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

Definition at line 630 of file ios_base.h.

5.614.5.14 **template**<typename _CharT, typename _Traits > **basic_ostream**< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::flush () [inherited]

Synchronizing the stream buffer.

Returns

*this

If `rdbuf()` is a null pointer, changes nothing.

Otherwise, calls `rdbuf() ->pubsync()`, and if that returns -1, sets `badbit`.

Definition at line 211 of file ostream.tcc.

References std::ios_base::badbit, std::ios_base::goodbit, std::basic_ios< _CharT, _Traits >::rdbuf(), std::basic_ios< _CharT, _Traits >::setstate(), and std::basic_ostream< _CharT, _Traits >::tellp().

Referenced by std::operator<<(), and std::basic_ostream< _CharT, _Traits >::write().

5.614.5.15 `template<typename _CharT, typename _Traits> streamsize std::basic_istream< _CharT, _Traits >::gcount ()`
`const [inline],[inherited]`

Character counting.

Returns

The number of characters extracted by the previous unformatted input function dispatched for this stream.

Definition at line 269 of file istream.

5.614.5.16 `template<typename _CharT , typename _Traits > basic_istream< _CharT, _Traits >::int_type`
`std::basic_istream< _CharT, _Traits >::get (void) [inherited]`

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns traits::eof().

Definition at line 236 of file istream.tcc.

References `std::basic_istream< _CharT, _Traits >::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

Referenced by `std::basic_istream< _CharT, _Traits >::get()`, and `std::basic_istream< _CharT, _Traits >::operator>>()`.

5.614.5.17 `template<typename _CharT , typename _Traits > basic_istream< _CharT, _Traits > & std::basic_istream<`
`_CharT, _Traits >::get (char_type & __c) [inherited]`

Simple extraction.

Parameters

<code>__c</code>	The character in which to store data.
<code>__c</code>	

Returns

*this

Tries to extract a character and store it in `__c`. If none are available, sets failbit and returns traits::eof().

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 272 of file istream.tcc.

References std::basic_istream< _CharT, _Traits >::_M_gcount, std::ios_base::badbit, std::ios_base::eofbit, std::ios_base::failbit, std::basic_istream< _CharT, _Traits >::get(), std::ios_base::goodbit, std::basic_ios< _CharT, _Traits >::rdbuf(), and std::basic_ios< _CharT, _Traits >::setstate().

5.614.5.18 template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::get (char_type * __s, streamsize __n, char_type __delim) [inherited]

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
<code>__n</code>	Maximum number of characters to store in <code>__s</code> .
<code>__delim</code>	A "stop" character.

Returns

*this

Characters are extracted and stored into `__s` until one of the following happens:

- `__n-1` characters are stored
- the input sequence reaches EOF
- the next character equals `__delim`, in which case the character is not extracted

If no characters are stored, failbit is set in the stream's error state.

In any case, a null character is stored into the next location in the array.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 309 of file istream.tcc.

References std::basic_istream< _CharT, _Traits >::_M_gcount, std::ios_base::badbit, std::ios_base::eofbit, std::ios_base::failbit, std::basic_istream< _CharT, _Traits >::get(), std::ios_base::goodbit, std::basic_ios< _CharT, _Traits >::rdbuf(), std::basic_ios< _CharT, _Traits >::setstate(), std::basic_streambuf< _CharT, _Traits >::sgetc(), and std::basic_streambuf< _CharT, _Traits >::snextc().

5.614.5.19 template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits >::get (char_type * __s, streamsize __n) [inline], [inherited]

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
<code>__n</code>	Maximum number of characters to store in <code>s</code> .

Returns

`*this`

Returns `get(__s,__n,widen("\n"))`.

Definition at line 354 of file `istream`.

5.614.5.20 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::get (__streambuf_type & __sb, char_type __delim) [inherited]`

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
<code>__delim</code>	A "stop" character.

Returns

`*this`

Characters are extracted and inserted into `__sb` until one of the following happens:

- the input sequence reaches EOF
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted)
- the next character equals `__delim` (in this case, the character is not extracted)
- an exception occurs (and in this case is caught)

If no characters are stored, failbit is set in the stream's error state.

Definition at line 356 of file `istream.tcc`.

References `std::basic_istream< _CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::basic_istream< _CharT, _Traits>::getline()`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, `std::basic_ios< _CharT, _Traits>::setstate()`, `std::basic_streambuf< _CharT, _Traits>::sgetc()`, `std::basic_streambuf< _CharT, _Traits>::snextc()`, and `std::basic_streambuf< _CharT, _Traits>::putc()`.

5.614.5.21 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::get (__streambuf_type & __sb) [inline],[inherited]`

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
-------------------	-------------------------------------

Returns

*this

Returns `get(__sb,widen("\n"))`.

Definition at line 387 of file istream.

5.614.5.22 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::getline (char_type * __s, streamsize __n, char_type __delim) [inherited]`

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.
<code>__delim</code>	A "stop" character.

Returns

*this

Extracts and stores characters into `__s` until one of the following happens. Note that these criteria are required to be tested in the order listed here, to allow an input line to exactly fill the `__s` array without setting failbit.

1. the input sequence reaches end-of-file, in which case eofbit is set in the stream error state
2. the next character equals `__delim`, in which case the character is extracted (and therefore counted in `gcount()`) but not stored
3. `__n-1` characters are stored, in which case failbit is set in the stream error state

If no characters are extracted, failbit is set. (An empty line of input should therefore not cause failbit to be set.)

In any case, a null character is stored in the next location in the array.

Definition at line 400 of file istream.tcc.

References `std::basic_istream< _CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_istream< _CharT, _Traits>::ignore()`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, `std::basic_streambuf< _CharT, _Traits>::sbumpc()`, `std::basic_ios< _CharT, _Traits>::setstate()`, `std::basic_streambuf< _CharT, _Traits>::sgetc()`, and `std::basic_streambuf< _CharT, _Traits>::snextc()`.

Referenced by `std::basic_istream< _CharT, _Traits>::get()`.

5.614.5.23 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::getline (char_type * __s, streamsize __n) [inline], [inherited]`

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.

Returns

`*this`

Returns `getline(__s,__n,widen("\n"))`.

Definition at line 427 of file `istream`.

5.614.5.24 `template<> basic_istream< char > & std::basic_istream< char >::getline (char_type * __s, streamsize __n, char_type __delim) [inherited]`

Explicit specialization declarations, defined in `src/istream.cc`.

5.614.5.25 `locale std::ios_base::getloc () const [inline],[inherited]`

Locale access.

Returns

A copy of the current locale.

If `imbue(loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale()`, the global C++ locale.

Definition at line 763 of file `ios_base.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::money_put< _CharT, _Outlter >::do_put()`, `std::operator>>()`, and `std::ws()`.

5.614.5.26 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::good () const [inline],[inherited]`

Fast error checking.

Returns

True if no error flags are set.

A wrapper around `rdstate`.

Definition at line 180 of file `basic_ios.h`.

Referenced by `std::__detail::operator>>()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_istream< _CharT, _Traits >::sentry::sentry()`.

5.614.5.27 `template<typename _CharT, typename _Traits > basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::ignore (streamsize __n, int_type __delim) [inherited]`

Discarding characters.

Parameters

<code>__n</code>	Number of characters to discard.
<code>__delim</code>	A "stop" character.

Returns

*this

Extracts characters and throws them away until one of the following happens:

- if `__n != std::numeric_limits<int>::max()`, `__n` characters are extracted
- the input sequence reaches end-of-file
- the next character equals `__delim` (in this case, the character is extracted); note that this condition will never occur if `__delim` equals `traits::eof()`.

NB: Provide three overloads, instead of the single function (with defaults) mandated by the Standard: this leads to a better performing implementation, while still conforming to the Standard.

Definition at line 555 of file istream.tcc.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_streambuf< _CharT, _Traits >::sbumpc()`, `std::basic_ios< _CharT, _Traits >::setstate()`, `std::basic_streambuf< _CharT, _Traits >::sgetc()`, and `std::basic_streambuf< _CharT, _Traits >::snextc()`.

5.614.5.28 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::ignore (streamsize __n) [inherited]`

Simple extraction.

Returns

A character, or `eof()`.

Tries to extract a character. If none are available, sets failbit and returns `traits::eof()`.

Definition at line 493 of file istream.tcc.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_ios< _CharT, _Traits >::setstate()`, `std::basic_streambuf< _CharT, _Traits >::sgetc()`, and `std::basic_streambuf< _CharT, _Traits >::snextc()`.

5.614.5.29 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::ignore (void) [inherited]`

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns traits::eof().

Definition at line 460 of file istream.tcc.

References `std::basic_istream< _CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, `std::basic_streambuf< _CharT, _Traits>::sbumpc()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream< _CharT, _Traits>::getline()`, and `std::basic_istream< _CharT, _Traits>::ignore()`.

5.614.5.30 `template<typename _CharT, typename _Traits> locale std::basic_ios< _CharT, _Traits>::imbue (const locale & __loc) [inherited]`

Moves to a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls `ios_base::imbue(loc)`, and if a stream buffer is associated with this stream, calls that buffer's `pubimbue(loc)`.

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Definition at line 114 of file basic_ios.tcc.

References `std::ios_base::imbue()`.

Referenced by `std::basic_ios< char, _Traits>::fill()`.

5.614.5.31 `template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits>::init (basic_streambuf< _CharT, _Traits> * __sb) [protected], [inherited]`

All setup is performed here.

This is called from the public constructor. It is not virtual and cannot be redefined.

Definition at line 126 of file basic_ios.tcc.

Referenced by `std::basic_ios< char, _Traits>::basic_ios()`.

5.614.5.32 long& std::ios_base::iword (int __ix) [inline],[inherited]

Access to integer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to an integer associated with the index.

The iword function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use xalloc to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 809 of file ios_base.h.

5.614.5.33 template<typename _CharT, typename _Traits> char std::basic_ios<_CharT,_Traits>::narrow (char_type __c, char __dfault) const [inline],[inherited]

Squeezes characters.

Parameters

<code>__c</code>	The character to narrow.
<code>__dfault</code>	The character to narrow.

Returns

The narrowed character.

Maps a character of `char_type` to a character of `char`, if possible.

Returns the result of

```
1 std::use_facet<ctype<char_type>> >(getloc()).narrow(c,dfault)
```

Additional l10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Definition at line 430 of file basic_ios.h.

5.614.5.34 `template<typename _CharT, typename _Traits> std::basic_ios<_CharT, _Traits>::operator bool () const`
`[inline], [explicit], [inherited]`

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`.

Definition at line 117 of file `basic_ios.h`.

5.614.5.35 `template<typename _CharT, typename _Traits> bool std::basic_ios<_CharT, _Traits>::operator! () const`
`[inline], [inherited]`

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`.

Definition at line 125 of file `basic_ios.h`.

5.614.5.36 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(__ostream_type &)(__ostream_type & __pf)` `[inline], [inherited]`

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

Definition at line 108 of file `ostream`.

Referenced by `std::basic_ostream<_CharT, _Traits>::operator<<()`, `std::operator<<()`, and `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`.

5.614.5.37 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(__ios_type &)(__ios_type & __pf)` `[inline], [inherited]`

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

Definition at line 117 of file `ostream`.

5.614.5.38 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(ios_base &)(ios_base & __pf)` `[inline], [inherited]`

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

Definition at line 127 of file `ostream`.

5.614.5.39 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(long __n)` `[inline], [inherited]`

Integer arithmetic inserters.

Parameters

<code>_↔</code> <code>_n</code>	A variable of builtin integral type.
------------------------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 166 of file ostream.

5.614.5.40 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits >::operator<<(unsigned long __n) [inline], [inherited]`

Integer arithmetic inserters.

Parameters

<code>_↔</code> <code>_n</code>	A variable of builtin integral type.
------------------------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 170 of file ostream.

5.614.5.41 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits >::operator<<(bool __n) [inline], [inherited]`

Integer arithmetic inserters.

Parameters

<code>_↔</code> <code>_n</code>	A variable of builtin integral type.
------------------------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 174 of file ostream.

5.614.5.42 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::operator<<(short __n) [inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 92 of file `ostream.tcc`.

References `std::ios_base::basefield`, `std::ios_base::flags()`, `std::ios_base::hex`, `std::ios_base::oct`, and `std::basic_ostream<_CharT, _Traits>::operator<<()`.

5.614.5.43 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(unsigned short __n) [inline], [inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 181 of file `ostream`.

5.614.5.44 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::operator<<(int __n) [inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 106 of file `ostream.tcc`.

References `std::ios_base::basefield`, `std::ios_base::flags()`, `std::ios_base::hex`, `std::ios_base::oct`, and `std::basic_ostream< _CharT, _Traits >::operator<<()`.

5.614.5.45 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits >::operator<< (unsigned int __n) [inline], [inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 192 of file `ostream`.

5.614.5.46 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits >::operator<< (long long __n) [inline], [inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 201 of file `ostream`.

5.614.5.47 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(unsigned long long __n) [inline], [inherited]`

Integer arithmetic inserters.

Parameters

\leftrightarrow _n	A variable of builtin integral type.
-------------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 205 of file ostream.

5.614.5.48 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(double __f) [inline], [inherited]`

Floating point arithmetic inserters.

Parameters

\leftrightarrow \leftrightarrow \leftrightarrow \leftrightarrow f	A variable of builtin floating point type.
---	--

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 220 of file ostream.

5.614.5.49 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(float __f) [inline], [inherited]`

Floating point arithmetic inserters.

Parameters

\leftrightarrow \leftrightarrow \leftrightarrow \leftrightarrow f	A variable of builtin floating point type.
---	--

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 224 of file `ostream`.

5.614.5.50 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(long double __f) [inline],[inherited]`

Floating point arithmetic inserters.

Parameters

<code>↔</code>	A variable of builtin floating point type.
<code>↔</code>	
<code>↔</code>	
<code>↔</code>	
<code>f</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 232 of file `ostream`.

5.614.5.51 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(const void *__p) [inline],[inherited]`

Pointer arithmetic inserters.

Parameters

<code>↔</code>	A variable of pointer type.
<code>p</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 245 of file `ostream`.

5.614.5.52 `template<typename _CharT, typename _Traits> basic_ostream<_CharT,_Traits> & std::basic_ostream<_CharT,_Traits>::operator<<(__streambuf_type * __sb)` [inherited]

Extracting from another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is NULL, the stream will set failbit in its error state.

Characters are extracted from `__sb` and inserted into `*this` until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output sequence fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs while getting a character from `__sb`, which sets failbit in the error state

If the function inserts no characters, failbit is set.

Definition at line 120 of file ostream.tcc.

References `std::ios_base::badbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ostream<_CharT, _Traits>::put()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

```
5.614.5.53 template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>
::operator>>( __istream_type &(*)(__istream_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `iomanip` header.

Definition at line 120 of file istream.

Referenced by `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::basic_istream<_CharT, _Traits>::sentry`, `std::sentry`, and `std::ws`.

```
5.614.5.54 template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>
::operator>>( __ios_type &(*)(__ios_type &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `iomanip` header.

Definition at line 124 of file istream.

```
5.614.5.55 template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>
::operator>>( ios_base &(*)(ios_base &) __pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `iomanip` header.

Definition at line 131 of file istream.

```
5.614.5.56 template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>
::operator>>( bool &__n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

<code>__↔ __n</code>	A variable of builtin integral type.
--------------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 168 of file `istream`.

5.614.5.57 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::operator>> (short & __n) [inherited]`

Integer arithmetic extractors.

Parameters

<code>__↔ __n</code>	A variable of builtin integral type.
--------------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 114 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::failbit`, `std::num_get< _CharT, _InIter >::get()`, `std::ios_base::goodbit`, `std::basic_istream< _CharT, _Traits >::operator>>()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

5.614.5.58 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits >::operator>> (unsigned short & __n) [inline], [inherited]`

Integer arithmetic extractors.

Parameters

<code>__↔ __n</code>	A variable of builtin integral type.
--------------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 175 of file `istream`.

5.614.5.59 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::operator>> (int & __n) [inherited]`

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 159 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::failbit`, `std::num_get<_CharT, _InIter>::get()`, `std::ios_base::goodbit`, `std::basic_istream<_CharT, _Traits>::operator>>()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

5.614.5.60 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (unsigned int & __n) [inline], [inherited]`

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 182 of file `istream`.

```
5.614.5.61 template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> ( long & __n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

<code>_↔</code> <code>_n</code>	A variable of builtin integral type.
------------------------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 186 of file `istream`.

```
5.614.5.62  template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits
              >::operator>> ( unsigned long & _n )  [inline],[inherited]
```

Integer arithmetic extractors.

Parameters

<code>_↔</code> <code>_n</code>	A variable of builtin integral type.
------------------------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 190 of file `istream`.

```
5.614.5.63  template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits
              >::operator>> ( long long & _n )  [inline],[inherited]
```

Integer arithmetic extractors.

Parameters

<code>_↔</code> <code>_n</code>	A variable of builtin integral type.
------------------------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 195 of file `istream`.

5.614.5.64 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (unsigned long long &__n) [inline], [inherited]`

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 199 of file `istream`.

5.614.5.65 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (float &__f) [inline], [inherited]`

Floating point arithmetic extractors.

Parameters

<code>__f</code>	A variable of builtin floating point type.
------------------	--

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 214 of file `istream`.

5.614.5.66 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (double &__f) [inline], [inherited]`

Floating point arithmetic extractors.

Parameters

↔	A variable of builtin floating point type.
↔	
↔	
↔	
<i>f</i>	

Returns

**this* if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 218 of file `istream`.

5.614.5.67 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::operator>> (long double &__f) [inline], [inherited]`

Floating point arithmetic extractors.

Parameters

↔	A variable of builtin floating point type.
↔	
↔	
↔	
<i>f</i>	

Returns

**this* if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 222 of file `istream`.

5.614.5.68 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::operator>> (void *&__p) [inline], [inherited]`

Basic arithmetic extractors.

Parameters

↔	A variable of pointer type.
<i>p</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 235 of file `istream`.

5.614.5.69 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::operator>> (__streambuf_type * __sb)` [inherited]

Extracting into another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is NULL, the stream will set failbit in its error state.

Characters are extracted from this stream and inserted into the `__sb` streambuf until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs (and in this case is caught)

If the function inserts no characters, failbit is set.

Definition at line 204 of file `istream.tcc`.

References `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::basic_istream< _CharT, _Traits>::get()`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

5.614.5.70 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits>::int_type std::basic_istream< _CharT, _Traits>::peek (void)` [inherited]

Looking ahead in the stream.

Returns

The next character, or `eof()`.

If, after constructing the sentry object, `good()` is false, returns `traits::eof()`. Otherwise reads but does not extract the next input character.

Definition at line 620 of file `istream.tcc`.

References `std::basic_istream< _CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, `std::basic_istream< _CharT, _Traits>::read()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream< _CharT, _Traits>::ignore()`.

5.614.5.71 `streamsize std::ios_base::precision () const` `[inline],[inherited]`

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Definition at line 689 of file `ios_base.h`.

Referenced by `std::basic_ios<_CharT,_Traits>::copyfmt()`, `std::num_get<_CharT,_InIter>::do_get()`, `std::normal_distribution<_RealType>::operator()()`, `std::gamma_distribution<_RealType>::operator()()`, `std::negative_binomial_distribution<_IntType>::operator()()`, and `std::operator>>()`.

5.614.5.72 `streamsize std::ios_base::precision (streamsize __prec)` `[inline],[inherited]`

Changing flags.

Parameters

<code>__prec</code>	The new precision value.
---------------------	--------------------------

Returns

The previous value of `precision()`.

Definition at line 698 of file `ios_base.h`.

5.614.5.73 `template<typename _CharT,typename _Traits> basic_ostream<_CharT,_Traits> & std::basic_ostream<_CharT,_Traits>::put (char_type __c)` `[inherited]`

Simple insertion.

Parameters

<code>__c</code>	The character to insert.
------------------	--------------------------

Returns

`*this`

Tries to insert `__c`.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 149 of file ostream.tcc.

References std::ios_base::badbit, std::ios_base::goodbit, std::basic_ios< _CharT, _Traits >::rdbuf(), std::basic_ios< _CharT, _Traits >::setstate(), and std::basic_ostream< _CharT, _Traits >::write().

Referenced by std::basic_ostream< _CharT, _Traits >::operator<<().

5.614.5.74 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::putback(char_type __c) [inherited]`

Unextracting a single character.

Parameters

<code>__c</code>	The character to push back into the input stream.
------------------	---

Returns

*this

If rdbuf() is not null, calls rdbuf() -> sputbackc(c).

If rdbuf() is null or if sputbackc() fails, sets badbit in the error state.

Note

This function first clears eofbit. Since no characters are extracted, the next call to gcount() will return 0, as required by DR 60.

Definition at line 711 of file istream.tcc.

References std::basic_istream< _CharT, _Traits >::M_gcount, std::ios_base::badbit, std::basic_ios< _CharT, _Traits >::clear(), std::ios_base::eofbit, std::ios_base::goodbit, std::basic_ios< _CharT, _Traits >::rdbuf(), std::basic_ios< _CharT, _Traits >::rdstate(), std::basic_ios< _CharT, _Traits >::setstate(), std::basic_streambuf< _CharT, _Traits >::sputbackc(), and std::basic_istream< _CharT, _Traits >::unget().

Referenced by std::basic_istream< _CharT, _Traits >::readsome().

5.614.5.75 `void*& std::ios_base::pword(int __ix) [inline], [inherited]`

Access to void pointer array.

Parameters

<code>_↵ _ix</code>	Index into the array.
-------------------------	-----------------------

Returns

A reference to a `void*` associated with the index.

The `pwd` function provides access to an array of pointers that can be used for any purpose. The array grows as required to hold the supplied index. All pointers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 830 of file `ios_base.h`.

```
5.614.5.76  template<typename _CharT, typename _Traits> basic_streambuf<_CharT, _Traits>* std::basic_ios<_CharT,
           _Traits>::rdbuf( ) const    [inline],[inherited]
```

Accessing the underlying buffer.

Returns

The current stream buffer.

This does not change the state of the stream.

Definition at line 321 of file `basic_ios.h`.

Referenced by `std::basic_ostream<_CharT, _Traits>::flush()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_ostream<_CharT, _Traits>::operator<<()`, `std::tr2::operator>>()`, `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::operator>>()`, `std::basic_istream<_CharT, _Traits>::peek()`, `std::basic_ostream<_CharT, _Traits>::put()`, `std::basic_istream<_CharT, _Traits>::putback()`, `std::basic_ios<char, _Traits>::rdbuf()`, `std::basic_istream<_CharT, _Traits>::read()`, `std::basic_istream<_CharT, _Traits>::readsome()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_ostream<_CharT, _Traits>::seekp()`, `std::basic_istream<_CharT, _Traits>::sentry()`, `std::sentry()`, `std::basic_istream<_CharT, _Traits>::sync()`, `std::basic_istream<_CharT, _Traits>::tellg()`, `std::basic_ostream<_CharT, _Traits>::tellp()`, `std::basic_istream<_CharT, _Traits>::unget()`, and `std::ws()`.

```
5.614.5.77  template<typename _CharT, typename _Traits> basic_streambuf<_CharT, _Traits>* std::basic_ios<_CharT,
           _Traits>::rdbuf( basic_streambuf<_CharT, _Traits>* __sb )    [inherited]
```

Changing the underlying buffer.

Parameters

<code>__sb</code>	The new stream buffer.
-------------------	------------------------

Returns

The previous stream buffer.

Associates a new buffer with the current stream, and clears the error state.

Due to historical accidents which the LWG refuses to correct, the I/O library suffers from a design error: this function is hidden in derived classes by overrides of the zero-argument `rdbuf()`, which is non-virtual for hysterical raisins. As a result, you must use explicit qualifications to access this function via any derived class. For example:

```
1 std::fstream    foo;           // or some other derived type
2 std::streambuf* p = .....;
3
4 foo.ios::rdbuf(p);           // ios == basic_ios<char>
```

Definition at line 53 of file `basic_ios.tcc`.

5.614.5.78 `template<typename _CharT, typename _Traits> iosstate std::basic_ios< _CharT, _Traits >::rdstate () const`
`[inline], [inherited]`

Returns the error state of the stream buffer.

Returns

A bit pattern (well, isn't everything?)

See `std::ios_base::iosstate` for the possible bit values. Most users will call one of the interpreting wrappers, e.g., `good()`.

Definition at line 137 of file `basic_ios.h`.

Referenced by `std::basic_ios< char, _Traits >::bad()`, `std::basic_ios< char, _Traits >::eof()`, `std::basic_ios< char, _Traits >::fail()`, `std::basic_ios< char, _Traits >::good()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ios< char, _Traits >::setstate()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

5.614.5.79 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::read (char_type * __s, streamsize __n)` `[inherited]`

Extraction without delimiters.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

*this

If the stream state is `good()`, extracts characters and stores them into `__s` until one of the following happens:

- `__n` characters are stored
- the input sequence reaches end-of-file, in which case the error state is set to `failbit|eofbit`.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 650 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_istream<_CharT, _Traits>::readsome()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream<_CharT, _Traits>::peek()`.

5.614.5.80 `template<typename _CharT, typename _Traits> streamsize std::basic_istream<_CharT, _Traits>::readsome (char_type* __s, streamsize __n) [inherited]`

Extraction until the buffer is exhausted, but no more.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

The number of characters extracted.

Extracts characters and stores them into `__s` depending on the number of characters remaining in the streambuf's buffer, `rdbuf()->in_avail()`, called `A` here:

- if `A == -1`, sets `eofbit` and extracts no characters
- if `A == 0`, extracts no characters
- if `A > 0`, extracts `min(A, n)`

The goal is to empty the current buffer, and to not request any more from the external input sequence controlled by the streambuf.

Definition at line 679 of file istream.tcc.

References std::basic_istream< _CharT, _Traits >::_M_gcount, std::ios_base::badbit, std::ios_base::eofbit, std::ios_base::goodbit, std::min(), std::basic_istream< _CharT, _Traits >::putback(), std::basic_ios< _CharT, _Traits >::rdbuf(), and std::basic_ios< _CharT, _Traits >::setstate().

Referenced by std::basic_istream< _CharT, _Traits >::read().

5.614.5.81 void std::ios_base::register_callback (event_callback __fn, int __index) [inherited]

Add the callback __fn with parameter __index.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

5.614.5.82 template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::seekg (pos_type __pos) [inherited]

Changing the current read position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

*this

If fail() is not true, calls rdbuf()->pubseekpos(__pos). If that function fails, sets failbit.

Note

This function first clears eofbit. It does not count the number of characters extracted, if any, and therefore does not affect the next call to gcount().

Definition at line 845 of file istream.tcc.

References std::ios_base::badbit, std::basic_ios< _CharT, _Traits >::clear(), std::ios_base::eofbit, std::basic_ios< _CharT, _Traits >::fail(), std::ios_base::failbit, std::ios_base::goodbit, std::ios_base::in, std::basic_ios< _CharT, _Traits >::rdbuf(), std::basic_ios< _CharT, _Traits >::rdstate(), and std::basic_ios< _CharT, _Traits >::setstate().

Referenced by std::basic_istream< _CharT, _Traits >::tellg().

5.614.5.83 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::seekg (off_type __off, ios_base::seekdir __dir) [inherited]`

Changing the current read position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

`*this`

If `fail()` is not true, calls `rdbuf()->pubseekoff(__off, __dir)`. If that function fails, sets failbit.

Note

This function first clears eofbit. It does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

Definition at line 884 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::basic_ios< _CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::basic_ios< _CharT, _Traits>::fail()`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::ios_base::in`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, `std::basic_ios< _CharT, _Traits>::rdstate()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

5.614.5.84 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::seekp (pos_type __pos) [inherited]`

Changing the current write position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

`*this`

If `fail()` is not true, calls `rdbuf()->pubseekpos(pos)`. If that function fails, sets failbit.

Definition at line 258 of file `ostream.tcc`.

References `std::ios_base::badbit`, `std::basic_ios< _CharT, _Traits>::fail()`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::ios_base::out`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

Referenced by `std::basic_ostream< _CharT, _Traits>::tellp()`.

5.614.5.85 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::seekp (off_type __off, ios_base::seekdir __dir)` [inherited]

Changing the current write position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

*this

If `fail()` is not true, calls `rdbuf()->pubseekoff(off, dir)`. If that function fails, sets failbit.

Definition at line 290 of file `ostream.tcc`.

References `std::ios_base::badbit`, `std::basic_ios< _CharT, _Traits>::fail()`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::ios_base::out`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

5.614.5.86 `fmtflags std::ios_base::setf (fmtflags __fmtfl)` [inline], [inherited]

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
----------------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set.

Definition at line 646 of file `ios_base.h`.

Referenced by `std::boolalpha()`, `std::dec()`, `std::fixed()`, `std::hex()`, `std::hexfloat()`, `std::left()`, `std::oct()`, `std::__detail<::operator>>()`, `std::right()`, `std::scientific()`, `std::showbase()`, `std::showpoint()`, `std::showpos()`, `std::skipws()`, `std::unitbuf()`, and `std::uppercase()`.

5.614.5.87 `fmtflags std::ios_base::setf (fmtflags __fmtfl, fmtflags __mask)` [inline], [inherited]

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
<code>__mask</code>	The flags mask for <code>__fmtfl</code> .

Returns

The previous format control flags.

This function clears *mask* in the format flags, then sets *fmtfl* & *mask*. An example mask is `ios_base::adjustfield`.

Definition at line 663 of file `ios_base.h`.

5.614.5.88 `template<typename _CharT, typename _Traits> void std::basic_ios<_CharT, _Traits>::setstate (iostate __state)`
`[inline], [inherited]`

Sets additional flags in the error state.

Parameters

<code>__state</code>	The additional state flag(s) to set.
----------------------	--------------------------------------

See `std::ios_base::iostate` for the possible bit values.

Definition at line 157 of file `basic_ios.h`.

Referenced by `std::basic_ostream<_CharT, _Traits>::flush()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_ostream<_CharT, _Traits>::operator<<()`, `std::operator<<()`, `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::tr2::operator>>()`, `std::operator>>()`, `std::basic_istream<_CharT, _Traits>::peek()`, `std::basic_ostream<_CharT, _Traits>::put()`, `std::basic_istream<_CharT, _Traits>::putback()`, `std::basic_istream<_CharT, _Traits>::read()`, `std::basic_istream<_CharT, _Traits>::readsome()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_ostream<_CharT, _Traits>::seekp()`, `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`, `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_istream<_CharT, _Traits>::sync()`, `std::basic_istream<_CharT, _Traits>::unset()`, and `std::ws()`.

5.614.5.89 `template<typename _CharT, typename _Traits> int std::basic_istream<_CharT, _Traits>::sync (void)`
`[inherited]`

Synchronizing the stream buffer.

Returns

0 on success, -1 on failure

If `rdbuf()` is a null pointer, returns -1.

Otherwise, calls `rdbuf() -> pubsync()`, and if that returns -1, sets `badbit` and returns -1.

Otherwise, returns 0.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

Definition at line 781 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::goodbit`, `std::basic_streambuf<_CharT, _Traits>::pubsync()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::setstate()`, and `std::basic_istream<_CharT, _Traits>::tellg()`.

Referenced by `std::basic_istream<_CharT, _Traits>::unset()`.

5.614.5.90 static bool std::ios_base::sync_with_stdio (bool __sync = true) [static],[inherited]

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., stdout) and the standard C++ objects (e.g., cout). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstream.html#std.io.filestreams.binary>

5.614.5.91 template<typename _CharT, typename _Traits > basic_istream< _CharT, _Traits >::pos_type
std::basic_istream< _CharT, _Traits >::tellg (void) [inherited]

Getting the current read position.

Returns

A file position object.

If fail() is not false, returns pos_type(-1) to indicate failure. Otherwise returns rdbuf()->pubseekoff(0, cur, in).

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to gcount(). At variance with putback, unget and seekg, eofbit is not cleared first.

Definition at line 817 of file istream.tcc.

References std::ios_base::badbit, std::ios_base::cur, std::basic_ios< _CharT, _Traits >::fail(), std::ios_base::in, std::basic_ios< _CharT, _Traits >::rdbuf(), and std::basic_istream< _CharT, _Traits >::seekg().

Referenced by std::basic_istream< _CharT, _Traits >::sync().

5.614.5.92 template<typename _CharT, typename _Traits > basic_ostream< _CharT, _Traits >::pos_type
std::basic_ostream< _CharT, _Traits >::tellp () [inherited]

Getting the current write position.

Returns

A file position object.

If fail() is not false, returns pos_type(-1) to indicate failure. Otherwise returns rdbuf()->pubseekoff(0, cur, out).

Definition at line 237 of file ostream.tcc.

References std::ios_base::badbit, std::ios_base::cur, std::basic_ios< _CharT, _Traits >::fail(), std::ios_base::out, std::basic_ios< _CharT, _Traits >::rdbuf(), and std::basic_ostream< _CharT, _Traits >::seekp().

Referenced by std::basic_ostream< _CharT, _Traits >::flush().

5.614.5.93 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>* std::basic_ios<_CharT, _Traits>::tie () const [inline], [inherited]`

Fetches the current *tied* stream.

Returns

A pointer to the tied stream, or NULL if the stream is not tied.

A stream may be *tied* (or synchronized) to a second output stream. When this stream performs any I/O, the tied stream is first flushed. For example, `std::cin` is tied to `std::cout`.

Definition at line 295 of file `basic_ios.h`.

Referenced by `std::basic_ios<char, _Traits>::basic_ios()`, `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`, and `std::basic_istream<_CharT, _Traits>::sentry::sentry()`.

5.614.5.94 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>* std::basic_ios<_CharT, _Traits>::tie (basic_ostream<_CharT, _Traits>* __tiestr) [inline], [inherited]`

Ties this stream to an output stream.

Parameters

<code>__tiestr</code>	The output stream.
-----------------------	--------------------

Returns

The previously tied output stream, or NULL if the stream was not tied.

This sets up a new tie; see `tie()` for more.

Definition at line 307 of file `basic_ios.h`.

5.614.5.95 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::unget (void) [inherited]`

Unextracting the previous character.

Returns

`*this`

If `rdbuf()` is not null, calls `rdbuf()->sungetc(c)`.

If `rdbuf()` is null or if `sungetc()` fails, sets `badbit` in the error state.

Note

This function first clears eofbit. Since no characters are extracted, the next call to `gcount()` will return 0, as required by DR 60.

Definition at line 746 of file `istream.tcc`.

References `std::basic_istream< _CharT, _Traits >::_M_gcount`, `std::ios_base::badbit`, `std::basic_ios< _CharT, _Traits >::clear()`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_ios< ↵_CharT, _Traits >::rdstate()`, `std::basic_ios< _CharT, _Traits >::setstate()`, `std::basic_streambuf< _CharT, _Traits >↵::sungetc()`, and `std::basic_istream< _CharT, _Traits >::sync()`.

Referenced by `std::__detail::operator>>()`, and `std::basic_istream< _CharT, _Traits >::putback()`.

5.614.5.96 `void std::ios_base::unsetf (fmtflags __mask)` `[inline]`, `[inherited]`

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

Definition at line 678 of file `ios_base.h`.

Referenced by `std::defaultfloat()`, `std::noboolalpha()`, `std::noshowbase()`, `std::noshowpoint()`, `std::noshowpos()`, `std↵::noskipws()`, `std::nounitbuf()`, and `std::nouppercase()`.

5.614.5.97 `template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::widen (char __c)`
`const` `[inline]`, `[inherited]`

Widens characters.

Parameters

<code>↵_c</code>	The character to widen.
------------------	-------------------------

Returns

The widened character.

Maps a character of `char` to a character of `char_type`.

Returns the result of

```
1 std::use_facet<ctype<char_type>> >(getloc()).widen(c)
```


Additional l10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Definition at line 449 of file `basic_ios.h`.

Referenced by `std::basic_ios<char, _Traits>::fill()`, `std::getline()`, `std::normal_distribution<_RealType>::operator()`, `std::gamma_distribution<_RealType>::operator()`, `std::negative_binomial_distribution<_IntType>::operator()`, `std::tr2::operator>>()`, and `std::operator>>()`.

5.614.5.98 `streamsize std::ios_base::width () const` `[inline]`, `[inherited]`

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Definition at line 712 of file `ios_base.h`.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::money_get<_CharT, _InIter>::do_get()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_put<_CharT, _OutIter>::do_put()`, and `std::operator>>()`.

5.614.5.99 `streamsize std::ios_base::width (streamsize __wide)` `[inline]`, `[inherited]`

Changing flags.

Parameters

<code>__wide</code>	The new width value.
---------------------	----------------------

Returns

The previous value of `width()`.

Definition at line 721 of file `ios_base.h`.

5.614.5.100 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::write (const char_type * __s, streamsize __n)` `[inherited]`

Character string insertion.

Parameters

<code>__s</code>	The array to insert.
<code>__n</code>	Maximum number of characters to insert.

Returns

*this

Characters are copied from ___s and inserted into the stream until one of the following happens:

- ___n characters are inserted
- inserting into the output sequence fails (in this case, badbit will be set in the stream's error state)

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 183 of file ostream.tcc.

References std::basic_ostream< _CharT, _Traits >::_M_write(), std::ios_base::badbit, and std::basic_ostream< _CharT, _Traits >::flush().

Referenced by std::basic_ostream< _CharT, _Traits >::put().

5.614.5.101 static int std::ios_base::xalloc () throw [static], [inherited]

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the iword and pword functions. The expectation is that an application calls xalloc in order to obtain an index in the iword and pword arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. xalloc is guaranteed to return an index that is safe to use in the iword and pword arrays.

5.614.6 Member Data Documentation

5.614.6.1 template<typename _CharT, typename _Traits> streamsize std::basic_istream< _CharT, _Traits >::_M_gcount [protected], [inherited]

The number of characters extracted in the previous unformatted function; see gcount().

Definition at line 82 of file istream.

Referenced by std::basic_istream< _CharT, _Traits >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::basic_istream< _CharT, _Traits >::ignore(), std::basic_istream< _CharT, _Traits >::peek(), std::basic_istream< _CharT, _Traits >::putback(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::readsome(), and std::basic_istream< _CharT, _Traits >::unget().

5.614.6.2 `const fmtflags std::ios_base::adjustfield` `[static], [inherited]`

A mask of left|right|internal. Useful for the 2-arg form of `setf`.

Definition at line 378 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _Inlter >::do_get()`, `std::num_put< _CharT, _Outlter >::do_put()`, `std::internal()`, `std::left()`, and `std::right()`.

5.614.6.3 `const openmode std::ios_base::app` `[static], [inherited]`

Seek to end before each write.

Definition at line 432 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

5.614.6.4 `const openmode std::ios_base::ate` `[static], [inherited]`

Open and seek to end immediately after opening.

Definition at line 435 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::open()`.

5.614.6.5 `const iostate std::ios_base::badbit` `[static], [inherited]`

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Definition at line 402 of file `ios_base.h`.

Referenced by `std::basic_ios< char, _Traits >::bad()`, `std::basic_ios< char, _Traits >::fail()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsomewhat()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::basic_ostream< _CharT, _Traits >::write()`.

5.614.6.6 `const fmtflags std::ios_base::basefield` `[static], [inherited]`

A mask of dec|oct|hex. Useful for the 2-arg form of `setf`.

Definition at line 381 of file `ios_base.h`.

Referenced by `std::dec()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::num_put< _CharT, _Outlter >::do_put()`, `std::hex()`, `std::oct()`, and `std::basic_ostream< _CharT, _Traits >::operator<<()`.

5.614.6.7 const seekdir std::ios_base::beg [static],[inherited]

Request a seek relative to the beginning of the stream.

Definition at line 464 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::seekpos(), and __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync().

5.614.6.8 const openmode std::ios_base::binary [static],[inherited]

Perform input and output in binary mode (as opposed to text mode). This is probably not what you think it is; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>.

Definition at line 440 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::showmanyc().

5.614.6.9 const fmtflags std::ios_base::boolalpha [static],[inherited]

Insert/extract `bool` in alphabetic rather than numeric format.

Definition at line 326 of file ios_base.h.

Referenced by std::boolalpha(), std::num_get< _CharT, _InIter >::do_get(), std::num_put< _CharT, _OutIter >::do_put(), and std::noboolalpha().

5.614.6.10 const seekdir std::ios_base::cur [static],[inherited]

Request a seek relative to the current position within the sequence.

Definition at line 467 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::imbue(), std::basic_filebuf< _CharT, _Traits >::overflow(), std::basic_filebuf< _CharT, _Traits >::pbackfail(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff(), std::basic_filebuf< _CharT, _Traits >::seekoff(), __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync(), std::basic_istream< _CharT, _Traits >::tellg(), and std::basic_ostream< _CharT, _Traits >::tellp().

5.614.6.11 const fmtflags std::ios_base::dec [static],[inherited]

Converts integer input or generates integer output in decimal base.

Definition at line 329 of file ios_base.h.

Referenced by std::dec().

5.614.6.12 `const seekdir std::ios_base::end` `[static], [inherited]`

Request a seek relative to the current end of the sequence.

Definition at line 470 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::open()`, and `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`.

5.614.6.13 `const iostate std::ios_base::eofbit` `[static], [inherited]`

Indicates that an input operation reached the end of an input sequence.

Definition at line 405 of file `ios_base.h`.

Referenced by `std::time_get< _CharT, _Inlter >::do_get()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::time_get< _CharT, _Inlter >::do_get_date()`, `std::time_get< _CharT, _Inlter >::do_get_monthname()`, `std::time_get< _CharT, _Inlter >::do_get_time()`, `std::time_get< _CharT, _Inlter >::do_get_weekday()`, `std::time_get< _CharT, _Inlter >::do_get_year()`, `std::basic_ios< char, _Traits >::eof()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::time_get< _CharT, _Inlter >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::ws()`.

5.614.6.14 `const iostate std::ios_base::failbit` `[static], [inherited]`

Indicates that an input operation failed to read the expected characters, or that an output operation failed to generate the desired characters.

Definition at line 410 of file `ios_base.h`.

Referenced by `std::time_get< _CharT, _Inlter >::do_date_order()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::time_get< _CharT, _Inlter >::do_get_monthname()`, `std::time_get< _CharT, _Inlter >::do_get_weekday()`, `std::time_get< _CharT, _Inlter >::do_get_year()`, `std::basic_ios< char, _Traits >::fail()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::time_get< _CharT, _Inlter >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_istream< _CharT, _Traits >::sentry::sentry()`.

5.614.6.15 `const fmtflags std::ios_base::fixed` `[static], [inherited]`

Generate floating-point output in fixed-point notation.

Definition at line 332 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _Inlter >::do_get()`, `std::fixed()`, and `std::hexfloat()`.

5.614.6.16 const fmtflags std::ios_base::floatfield [static],[inherited]

A mask of scientific|fixed. Useful for the 2-arg form of `setf`.

Definition at line 384 of file `ios_base.h`.

Referenced by `std::defaultfloat()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::fixed()`, `std::hexfloat()`, and `std::scientific()`.

5.614.6.17 const iostate std::ios_base::goodbit [static],[inherited]

Indicates all is well.

Definition at line 413 of file `ios_base.h`.

Referenced by `std::time_get< _CharT, _Inlter >::do_date_order()`, `std::time_get< _CharT, _Inlter >::do_get()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::time_get< _CharT, _Inlter >::do_get_monthname()`, `std::time_get< _CharT, _Inlter >::do_get_weekday()`, `std::time_get< _CharT, _Inlter >::do_get_year()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::time_get< _CharT, _Inlter >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_ios< char, _Traits >::rdstate()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sync()`, and `std::basic_istream< _CharT, _Traits >::unset()`.

5.614.6.18 const fmtflags std::ios_base::hex [static],[inherited]

Converts integer input or generates integer output in hexadecimal base.

Definition at line 335 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _Inlter >::do_get()`, `std::num_put< _CharT, _Outlter >::do_put()`, `std::hex()`, and `std::basic_ostream< _CharT, _Traits >::operator<<()`.

5.614.6.19 const openmode std::ios_base::in [static],[inherited]

Open for input. Default for `ifstream` and `fstream`.

Definition at line 443 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos()`, `std::basic_filebuf< _CharT, _Traits >::showmanyc()`, `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::underflow()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

5.614.6.20 `const fmtflags std::ios_base::internal` `[static], [inherited]`

Adds fill characters at a designated internal point in certain generated output, or identical to `right` if no such point is designated.

Definition at line 340 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, and `std::ios_base::internal()`.

5.614.6.21 `const fmtflags std::ios_base::left` `[static], [inherited]`

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.)

Definition at line 344 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, and `std::ios_base::left()`.

5.614.6.22 `const fmtflags std::ios_base::oct` `[static], [inherited]`

Converts integer input or generates integer output in octal base.

Definition at line 347 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _InIter >::do_get()`, `std::oct()`, and `std::basic_ostream< _CharT, _Traits >::operator<<()`.

5.614.6.23 `const openmode std::ios_base::out` `[static], [inherited]`

Open for output. Default for `ofstream` and `fstream`.

Definition at line 446 of file `ios_base.h`.

Referenced by `std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow()`, `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::pbackfail()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos()`, `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

5.614.6.24 `const fmtflags std::ios_base::right` `[static], [inherited]`

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Definition at line 351 of file `ios_base.h`.

Referenced by `std::right()`.

5.614.6.25 `const fmtflags std::ios_base::scientific` `[static], [inherited]`

Generates floating-point output in scientific notation.

Definition at line 354 of file `ios_base.h`.

Referenced by `std::hexfloat()`, and `std::scientific()`.

5.614.6.26 `const fmtflags std::ios_base::showbase` `[static], [inherited]`

Generates a prefix indicating the numeric base of generated integer output.

Definition at line 358 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::noshowbase()`, and `std::showbase()`.

5.614.6.27 `const fmtflags std::ios_base::showpoint` `[static], [inherited]`

Generates a decimal-point character unconditionally in generated floating-point output.

Definition at line 362 of file `ios_base.h`.

Referenced by `std::noshowpoint()`, and `std::showpoint()`.

5.614.6.28 `const fmtflags std::ios_base::showpos` `[static], [inherited]`

Generates a + sign in non-negative generated numeric output.

Definition at line 365 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _InIter >::do_get()`, `std::noshowpos()`, and `std::showpos()`.

5.614.6.29 `const fmtflags std::ios_base::skipws` `[static], [inherited]`

Skips leading white space before certain input operations.

Definition at line 368 of file `ios_base.h`.

Referenced by `std::noskipws()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, and `std::skipws()`.

5.614.6.30 `const openmode std::ios_base::trunc` `[static], [inherited]`

Open for input. Default for `ofstream`.

Definition at line 449 of file `ios_base.h`.

5.614.6.31 `const fmtflags std::ios_base::unitbuf` `[static], [inherited]`

Flushes output after each output operation.

Definition at line 371 of file `ios_base.h`.

Referenced by `std::nunitbuf()`, and `std::unitbuf()`.

5.614.6.32 `const fmtflags std::ios_base::uppercase` `[static], [inherited]`

Replaces certain lowercase letters with their uppercase equivalents in generated output.

Definition at line 375 of file `ios_base.h`.

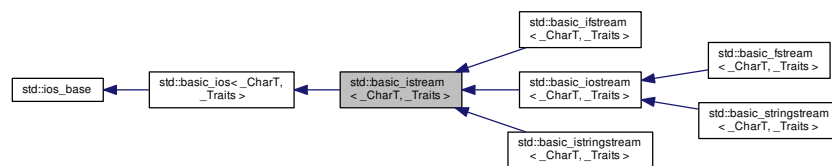
Referenced by `std::num_get< _CharT, _Inlter >::do_get()`, `std::num_put< _CharT, _Outlter >::do_put()`, `std::noupเปอร์case()`, and `std::uppercase()`.

The documentation for this class was generated from the following files:

- [iosfwd](#)
- [istream](#)

5.615 `std::basic_istream< _CharT, _Traits >` Class Template Reference

Inheritance diagram for `std::basic_istream< _CharT, _Traits >`:



Classes

- class [sentry](#)

Public Types

- typedef [ctype](#)< _CharT > **__ctype_type**
 - typedef [basic_ios](#)< _CharT, _Traits > **__ios_type**
 - typedef [basic_istream](#)< _CharT, _Traits > **__istream_type**
 - typedef [num_get](#)< _CharT, [istreambuf_iterator](#)< _CharT, _Traits > > **__num_get_type**
 - typedef [basic_streambuf](#)< _CharT, _Traits > **__streambuf_type**
 - typedef _CharT **char_type**
 - enum [event](#) { [erase_event](#), [imbue_event](#), [copyfmt_event](#) }
 - typedef void(* [event_callback](#)) (event __e, [ios_base](#) &__b, int __i)
 - typedef _ios_Fmtflags **fmtflags**
 - typedef _Traits::int_type **int_type**
 - typedef int **io_state**
 - typedef _ios_iostate **iostate**
 - typedef _Traits::off_type **off_type**
 - typedef int **open_mode**
 - typedef _ios_Openmode **openmode**
 - typedef _Traits::pos_type **pos_type**
 - typedef int **seek_dir**
 - typedef _ios_Seekdir **seekdir**
 - typedef [std::streamoff](#) **streamoff**
 - typedef [std::streampos](#) **streampos**
 - typedef _Traits **traits_type**
-
- typedef [num_put](#)< _CharT, [ostreambuf_iterator](#)< _CharT, _Traits > > **__num_put_type**

Public Member Functions

- [basic_istream](#) ([__streambuf_type](#) *__sb)
- virtual [~basic_istream](#) ()
- template<typename _ValueT >
[basic_istream](#)< _CharT, _Traits > & **_M_extract** (_ValueT &__v)
- const [locale](#) & **_M_getloc** () const
- void **_M_setstate** ([iostate](#) __state)
- bool **bad** () const
- void **clear** ([iostate](#) __state=[goodbit](#))
- [basic_ios](#) & **copyfmt** (const [basic_ios](#) &__rhs)
- bool **eof** () const
- [iostate](#) **exceptions** () const
- void **exceptions** ([iostate](#) __except)
- bool **fail** () const
- [char_type](#) **fill** () const
- [char_type](#) **fill** ([char_type](#) __ch)
- [fmtflags](#) **flags** () const
- [fmtflags](#) **flags** ([fmtflags](#) __fmtfl)
- [streamsize](#) **gcount** () const
- template<>
[basic_istream](#)< char > & **getline** ([char_type](#) *__s, [streamsize](#) __n, [char_type](#) __delim)
- template<>
[basic_istream](#)< wchar_t > & **getline** ([char_type](#) *__s, [streamsize](#) __n, [char_type](#) __delim)

- `locale getloc () const`
- `bool good () const`
- `template<>`
`basic_istream< char > & ignore (streamsize __n)`
- `template<>`
`basic_istream< char > & ignore (streamsize __n, int_type __delim)`
- `template<>`
`basic_istream< wchar_t > & ignore (streamsize __n)`
- `template<>`
`basic_istream< wchar_t > & ignore (streamsize __n, int_type __delim)`
- `locale imbue (const locale &__loc)`
- `long & iword (int __ix)`
- `char narrow (char_type __c, char __dfault) const`
- `__istream_type & operator>> (void *&__p)`
- `__istream_type & operator>> (__streambuf_type * __sb)`
- `streamsize precision () const`
- `streamsize precision (streamsize __prec)`
- `void *& pword (int __ix)`
- `basic_streambuf< _CharT, _Traits > * rdbuf () const`
- `basic_streambuf< _CharT, _Traits > * rdbuf (basic_streambuf< _CharT, _Traits > * __sb)`
- `iosstate rdstate () const`
- `void register_callback (event_callback __fn, int __index)`
- `fmtflags self (fmtflags __fmtfl)`
- `fmtflags self (fmtflags __fmtfl, fmtflags __mask)`
- `void setstate (iosstate __state)`
- `basic_ostream< _CharT, _Traits > * tie () const`
- `basic_ostream< _CharT, _Traits > * tie (basic_ostream< _CharT, _Traits > * __tiestr)`
- `void unsetf (fmtflags __mask)`
- `char_type widen (char __c) const`
- `streamsize width () const`
- `streamsize width (streamsize __wide)`
- `__istream_type & operator>> (__istream_type &(*__pf)(__istream_type &))`
- `__istream_type & operator>> (__ios_type &(*__pf)(__ios_type &))`
- `__istream_type & operator>> (ios_base &(*__pf)(ios_base &))`

Extractors

All the `operator>>` functions (aka formatted input functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (`noskipws`) set to `false`. This has several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to extract whatever data is appropriate for the type of the argument.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- `__istream_type & operator>> (bool &__n)`
- `__istream_type & operator>> (short &__n)`
- `__istream_type & operator>> (unsigned short &__n)`
- `__istream_type & operator>> (int &__n)`
- `__istream_type & operator>> (unsigned int &__n)`
- `__istream_type & operator>> (long &__n)`
- `__istream_type & operator>> (unsigned long &__n)`

- [__istream_type](#) & [operator>>](#) (long long &__n)
- [__istream_type](#) & [operator>>](#) (unsigned long long &__n)
- [__istream_type](#) & [operator>>](#) (float &__f)
- [__istream_type](#) & [operator>>](#) (double &__f)
- [__istream_type](#) & [operator>>](#) (long double &__f)

Unformatted Input Functions

All the unformatted input functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (`noskipws`) set to `true`. This has several effects, concluding with the setting of a status flag; see the `sentry` documentation for more.

If the `sentry` status is good, the function tries to extract whatever data is appropriate for the type of the argument.

The number of characters extracted is stored for later retrieval by `gcount()`.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- `int_type` [get](#) ()
- [__istream_type](#) & [get](#) (`char_type` &__c)
- [__istream_type](#) & [get](#) (`char_type` *__s, [streamsize](#) __n, `char_type` __delim)
- [__istream_type](#) & [get](#) (`char_type` *__s, [streamsize](#) __n)
- [__istream_type](#) & [get](#) ([__streambuf_type](#) &__sb, `char_type` __delim)
- [__istream_type](#) & [get](#) ([__streambuf_type](#) &__sb)
- [__istream_type](#) & [getline](#) (`char_type` *__s, [streamsize](#) __n, `char_type` __delim)
- [__istream_type](#) & [getline](#) (`char_type` *__s, [streamsize](#) __n)
- [__istream_type](#) & [ignore](#) ([streamsize](#) __n, `int_type` __delim)
- [__istream_type](#) & [ignore](#) ([streamsize](#) __n)
- [__istream_type](#) & [ignore](#) ()
- `int_type` [peek](#) ()
- [__istream_type](#) & [read](#) (`char_type` *__s, [streamsize](#) __n)
- [streamsize](#) [readsome](#) (`char_type` *__s, [streamsize](#) __n)
- [__istream_type](#) & [putback](#) (`char_type` __c)
- [__istream_type](#) & [unget](#) ()
- `int` [sync](#) ()
- `pos_type` [tellg](#) ()
- [__istream_type](#) & [seekg](#) (`pos_type`)
- [__istream_type](#) & [seekg](#) (`off_type`, `ios_base::seekdir`)
- [operator bool](#) () const
- `bool` [operator!](#) () const

Static Public Member Functions

- static `bool` [sync_with_stdio](#) (`bool` __sync=true)
- static `int` [xalloc](#) () throw ()

Static Public Attributes

- static const [fmtflags adjustfield](#)
- static const [openmode app](#)
- static const [openmode ate](#)
- static const [iosstate badbit](#)
- static const [fmtflags basefield](#)
- static const [seekdir beg](#)
- static const [openmode binary](#)
- static const [fmtflags boolalpha](#)
- static const [seekdir cur](#)
- static const [fmtflags dec](#)
- static const [seekdir end](#)
- static const [iosstate eofbit](#)
- static const [iosstate failbit](#)
- static const [fmtflags fixed](#)
- static const [fmtflags floatfield](#)
- static const [iosstate goodbit](#)
- static const [fmtflags hex](#)
- static const [openmode in](#)
- static const [fmtflags internal](#)
- static const [fmtflags left](#)
- static const [fmtflags oct](#)
- static const [openmode out](#)
- static const [fmtflags right](#)
- static const [fmtflags scientific](#)
- static const [fmtflags showbase](#)
- static const [fmtflags showpoint](#)
- static const [fmtflags showpos](#)
- static const [fmtflags skipws](#)
- static const [openmode trunc](#)
- static const [fmtflags unitbuf](#)
- static const [fmtflags uppercase](#)

Protected Types

- enum { [_S_local_word_size](#) }

Protected Member Functions

- **basic_istream** (const [basic_istream](#) &)=delete
- **basic_istream** ([basic_istream](#) &&__rhs)
- void **_M_cache_locale** (const [locale](#) &__loc)
- void **_M_call_callbacks** ([event](#) __ev) throw ()
- void **_M_dispose_callbacks** (void) throw ()
- template<typename _ValueT >
[_istream_type](#) & **_M_extract** (_ValueT &__v)
- [_Words](#) & **_M_grow_words** (int __index, bool __iword)
- void **_M_init** () throw ()

- void **_M_move** (ios_base &) noexcept
- void **_M_swap** (ios_base &__rhs) noexcept
- void **init** (basic_streambuf< _CharT, _Traits > *__sb)
- void **move** (basic_ios &__rhs)
- void **move** (basic_ios &&__rhs)
- **basic_istream** & **operator=** (const **basic_istream** &)=delete
- **basic_istream** & **operator=** (**basic_istream** &&__rhs)
- void **set_rdbuf** (basic_streambuf< _CharT, _Traits > *__sb)
- void **swap** (basic_ios &__rhs) noexcept
- void **swap** (**basic_istream** &__rhs)

Protected Attributes

- _Callback_list * **_M_callbacks**
- const __ctype_type * **_M_ctype**
- iostate **_M_exception**
- char_type **_M_fill**
- bool **_M_fill_init**
- fmtflags **_M_flags**
- streamsize **_M_gcount**
- locale **_M_ios_locale**
- _Words **_M_local_word** [_S_local_word_size]
- const __num_get_type * **_M_num_get**
- const __num_put_type * **_M_num_put**
- streamsize **_M_precision**
- basic_streambuf< _CharT, _Traits > * **_M_streambuf**
- iostate **_M_streambuf_state**
- basic_ostream< _CharT, _Traits > * **_M_tie**
- streamsize **_M_width**
- _Words * **_M_word**
- int **_M_word_size**
- _Words **_M_word_zero**

Friends

- class **sentry**

5.615.1 Detailed Description

```
template<typename _CharT, typename _Traits>
class std::basic_istream< _CharT, _Traits >
```

Template class basic_istream.

Template Parameters

<code>_CharT</code>	Type of character stream.
<code>_Traits</code>	Traits for character type, defaults to char_traits<_CharT>.

This is the base class for all input streams. It provides text formatting of all builtin types, and communicates with any class derived from `basic_streambuf` to do the actual input.

Definition at line 83 of file `iosfwd`.

5.615.2 Member Typedef Documentation

5.615.2.1 `template<typename _CharT, typename _Traits> typedef num_put<_CharT, ostreambuf_iterator<_CharT, _Traits>
> std::basic_ios<_CharT, _Traits>::__num_put_type [inherited]`

These are non-standard types.

Definition at line 89 of file `basic_ios.h`.

5.615.2.2 `typedef void(* std::ios_base::event_callback)(event __e, ios_base &__b, int __i) [inherited]`

The type of an event callback function.

Parameters

<code>__e</code>	One of the members of the event enum.
<code>__b</code>	Reference to the <code>ios_base</code> object.
<code>__i</code>	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several `ios_base` and `basic_ios` functions, specifically `imbue()`, `copyfmt()`, and `~ios()`.

Definition at line 504 of file `ios_base.h`.

5.615.2.3 `typedef _Ios_Fmtflags std::ios_base::fmtflags [inherited]`

This is a bitmask type.

`_Ios_Fmtflags` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `fmtflags` are:

- `boolalpha`
- `dec`
- `fixed`
- `hex`
- `internal`
- `left`

- `oct`
- `right`
- `scientific`
- `showbase`
- `showpoint`
- `showpos`
- `skipws`
- `unitbuf`
- `uppercase`
- `adjustfield`
- `basefield`
- `floatfield`

Definition at line 323 of file `ios_base.h`.

5.615.2.4 `typedef _ios_iostate std::ios_base::iostate` `[inherited]`

This is a bitmask type.

`_Ios_Iostate` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `iostate` are:

- `badbit`
- `eofbit`
- `failbit`
- `goodbit`

Definition at line 398 of file `ios_base.h`.

5.615.2.5 `typedef _ios_Openmode std::ios_base::openmode` `[inherited]`

This is a bitmask type.

`_Ios_Openmode` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `openmode` are:

- `app`
- `ate`
- `binary`
- `in`
- `out`
- `trunc`

Definition at line 429 of file `ios_base.h`.

5.615.2.6 `typedef _Ios_Seekdir std::ios_base::seekdir` [inherited]

This is an enumerated type.

`_Ios_Seekdir` is implementation-defined. Defined values of type `seekdir` are:

- `beg`
- `cur`, equivalent to `SEEK_CUR` in the C standard library.
- `end`, equivalent to `SEEK_END` in the C standard library.

Definition at line 461 of file `ios_base.h`.

5.615.3 Member Enumeration Documentation

5.615.3.1 `enum std::ios_base::event` [inherited]

The set of events that may be passed to an event callback.

`erase_event` is used during `~ios()` and `copyfmt()`. `imbue_event` is used during `imbue()`. `copyfmt_event` is used during `copyfmt()`.

Definition at line 487 of file `ios_base.h`.

5.615.4 Constructor & Destructor Documentation

5.615.4.1 `template<typename _CharT, typename _Traits> std::basic_istream<_CharT, _Traits>::basic_istream (__streambuf_type* __sb)` [inline], [explicit]

Base constructor.

This ctor is almost never called by the user directly, rather from derived classes' initialization lists, which pass a pointer to their own stream buffer.

Definition at line 93 of file `istream`.

5.615.4.2 `template<typename _CharT, typename _Traits> virtual std::basic_istream<_CharT, _Traits>::~~basic_istream ()` [inline], [virtual]

Base destructor.

This does very little apart from providing a virtual base dtor.

Definition at line 103 of file `istream`.

5.615.5 Member Function Documentation

5.615.5.1 const locale& std::ios_base::M_getloc () const [inline],[inherited]

Locale access.

Returns

A reference to the current locale.

Like getloc above, but returns a reference instead of generating a copy.

Definition at line 774 of file ios_base.h.

Referenced by std::time_get< _CharT, _Inlter >::do_date_order(), std::time_get< _CharT, _Inlter >::do_get(), std::money_get< _CharT, _Inlter >::do_get(), std::num_get< _CharT, _Inlter >::do_get(), std::time_get< _CharT, _Inlter >::do_get_date(), std::time_get< _CharT, _Inlter >::do_get_monthname(), std::time_get< _CharT, _Inlter >::do_get_time(), std::time_get< _CharT, _Inlter >::do_get_weekday(), std::time_get< _CharT, _Inlter >::do_get_year(), std::time_put< _CharT, _Outlter >::do_put(), std::num_put< _CharT, _Outlter >::do_put(), std::time_get< _CharT, _Inlter >::get(), and std::time_put< _CharT, _Outlter >::put().

5.615.5.2 template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::bad () const [inline],[inherited]

Fast error checking.

Returns

True if the badbit is set.

Note that other iostate flags may also be set.

Definition at line 211 of file basic_ios.h.

5.615.5.3 template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits >::clear (iostate __state = goodbit) [inherited]

[Re]sets the error state.

Parameters

<code>__state</code>	The new state flag(s) to set.
----------------------	-------------------------------

See std::ios_base::iostate for the possible bit values. Most users will not need to pass an argument.

Definition at line 41 of file basic_ios.tcc.

Referenced by std::basic_ios< char, _Traits >::exceptions(), std::__detail::operator>>(), std::basic_istream< _CharT, _Traits >::putback(), std::basic_ios< char, _Traits >::rdstate(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ios< char, _Traits >::setstate(), and std::basic_istream< _CharT, _Traits >::unset().

5.615.5.4 `template<typename _CharT, typename _Traits> basic_ios< _CharT, _Traits> & std::basic_ios< _CharT, _Traits>::copyfmt (const basic_ios< _CharT, _Traits> & __rhs) [inherited]`

Copies fields of `__rhs` into this.

Parameters

<code>__rhs</code>	The source values for the copies.
--------------------	-----------------------------------

Returns

Reference to this object.

All fields of `__rhs` are copied into this object except that `rdbuf()` and `rdstate()` remain unchanged. All values in the `pword` and `iword` arrays are copied. Before copying, each callback is invoked with `erase_event`. After copying, each (new) callback is invoked with `copyfmt_event`. The final step is to copy exceptions().

Definition at line 63 of file `basic_ios.tcc`.

References `std::basic_ios< _CharT, _Traits>::exceptions()`, `std::basic_ios< _CharT, _Traits>::fill()`, `std::ios_base::flags()`, `std::ios_base::getloc()`, `std::ios_base::precision()`, `std::basic_ios< _CharT, _Traits>::tie()`, `std::tie()`, and `std::ios_base::width()`.

Referenced by `std::basic_ios< char, _Traits>::rdbuf()`.

5.615.5.5 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits>::eof () const [inline], [inherited]`

Fast error checking.

Returns

True if the eofbit is set.

Note that other iostate flags may also be set.

Definition at line 190 of file `basic_ios.h`.

5.615.5.6 `template<typename _CharT, typename _Traits> iostate std::basic_ios< _CharT, _Traits>::exceptions () const [inline], [inherited]`

Throwing exceptions on errors.

Returns

The current exceptions mask.

This changes nothing in the stream. See the one-argument version of `exceptions(iostate)` for the meaning of the return value.

Definition at line 222 of file `basic_ios.h`.

Referenced by `std::basic_ios< _CharT, _Traits>::copyfmt()`, and `std::basic_ios< char, _Traits>::setstate()`.

5.615.5.7 `template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits>::exceptions (iostate __except) [inline], [inherited]`

Throwing exceptions on errors.

Parameters

<code>__except</code>	The new exceptions mask.
-----------------------	--------------------------

By default, error flags are set silently. You can set an exceptions mask for each stream; if a bit in the mask becomes set in the error flags, then an exception of type `std::ios_base::failure` is thrown.

If the error flag is already set when the exceptions mask is added, the exception is immediately thrown. Try running the following under GCC 3.1 or later:

```
1 #include <iostream>
2 #include <fstream>
3 #include <exception>
4
5 int main()
6 {
7     std::set_terminate (__gnu_cxx::__verbose_terminate_handler);
8
9     std::ifstream f ("/etc/motd");
10
11     std::cerr << "Setting badbit\n";
12     f.setstate (std::ios_base::badbit);
13
14     std::cerr << "Setting exception mask\n";
15     f.exceptions (std::ios_base::badbit);
16 }
```

Definition at line 257 of file `basic_ios.h`.

5.615.5.8 `template<typename _CharT, typename _Traits> bool std::basic_istream< _CharT, _Traits >::fail () const`
`[inline], [inherited]`

Fast error checking.

Returns

True if either the badbit or the failbit is set.

Checking the badbit in `fail()` is historical practice. Note that other iostate flags may also be set.

Definition at line 201 of file `basic_ios.h`.

Referenced by `std::basic_istream< char, _Traits >::operator bool()`, `std::basic_istream< char, _Traits >::operator!()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, and `std::regex_traits< _Ch_type >::value()`.

5.615.5.9 `template<typename _CharT, typename _Traits> char_type std::basic_istream< _CharT, _Traits >::fill () const`
`[inline], [inherited]`

Retrieves the *empty* character.

Returns

The current fill character.

It defaults to a space (' ') in the current locale.

Definition at line 370 of file `basic_ios.h`.

Referenced by `std::basic_istream< _CharT, _Traits >::copyfmt()`, `std::basic_istream< char, _Traits >::fill()`, `std::normal_distribution< _RealType >::operator()()`, `std::gamma_distribution< _RealType >::operator()()`, `std::negative_binomial_distribution< _IntType >::operator()()`, and `std::operator>>()`.

5.615.5.10 `template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::fill (char_type __ch) [inline],[inherited]`

Sets a new *empty* character.

Parameters

<code>__ch</code>	The new character.
-------------------	--------------------

Returns

The previous fill character.

The fill character is used to fill out space when P+ characters have been requested (e.g., via `setw`), Q characters are actually used, and Q<P. It defaults to a space (' ') in the current locale.

Definition at line 390 of file `basic_ios.h`.

5.615.5.11 `fmtflags std::ios_base::flags () const [inline],[inherited]`

Access to format flags.

Returns

The format control flags for both input and output.

Definition at line 619 of file `ios_base.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >::discard()`, `std::money_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::operator()()`, `std::discard_block_engine< _RandomNumberEngine, __p, __r >::operator()()`, `std::shuffle_order_engine< _RandomNumberEngine, __k >::operator()()`, `std::normal_distribution< _RealType >::operator()()`, `std::gamma_distribution< _RealType >::operator()()`, `std::binomial_distribution< _IntType >::operator()()`, `std::negative_binomial_distribution< _IntType >::operator()()`, `std::poisson_distribution< _IntType >::operator()()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, `std::__detail::operator>>()`, `std::operator>>()`, `std::linear_congruential_engine< _UIntType, __a, __c, __m >::seed()`, and `std::basic_istream< _CharT, _Traits >::sentry::sentry()`.

5.615.5.12 `fmtflags std::ios_base::flags (fmtflags __fmtfl) [inline],[inherited]`

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

Definition at line 630 of file ios_base.h.

5.615.5.13 `template<typename _CharT, typename _Traits> streamsize std::basic_istream< _CharT, _Traits >::gcount ()
const [inline]`

Character counting.

Returns

The number of characters extracted by the previous unformatted input function dispatched for this stream.

Definition at line 269 of file istream.

5.615.5.14 `template<typename _CharT , typename _Traits > basic_istream< _CharT, _Traits >::int_type
std::basic_istream< _CharT, _Traits >::get (void)`

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns traits::eof().

Definition at line 236 of file istream.tcc.

References std::basic_istream< _CharT, _Traits >::M_gcount, std::ios_base::badbit, std::ios_base::eofbit, std::ios_base::failbit, std::ios_base::goodbit, std::basic_ios< _CharT, _Traits >::rdbuf(), and std::basic_ios< _CharT, _Traits >::setstate().

Referenced by std::basic_istream< _CharT, _Traits >::get(), and std::basic_istream< _CharT, _Traits >::operator>>().

5.615.5.15 `template<typename _CharT, typename _Traits > basic_istream< _CharT, _Traits > & std::basic_istream<
_CharT, _Traits >::get (char_type & __c)`

Simple extraction.

Parameters

<code>__c</code>	The character in which to store data.
------------------	---------------------------------------

Returns

*this

Tries to extract a character and store it in `__c`. If none are available, sets failbit and returns `traits::eof()`.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 272 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::basic_istream<_CharT, _Traits>::get()`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

5.615.5.16 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::get (char_type * __s, streamsize __n, char_type __delim)`

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
<code>__n</code>	Maximum number of characters to store in <code>__s</code> .
<code>__delim</code>	A "stop" character.

Returns

*this

Characters are extracted and stored into `__s` until one of the following happens:

- `__n-1` characters are stored
- the input sequence reaches EOF
- the next character equals `__delim`, in which case the character is not extracted

If no characters are stored, failbit is set in the stream's error state.

In any case, a null character is stored into the next location in the array.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 309 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::basic_istream<_CharT, _Traits>::get()`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sgetc()`, and `std::basic_streambuf<_CharT, _Traits>::snextc()`.

5.615.5.17 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits >::get (char_type * __s, streamsize __n) [inline]`

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
<code>__n</code>	Maximum number of characters to store in <i>s</i> .

Returns

*this

Returns `get(__s,__n,widen("\n"))`.

Definition at line 354 of file istream.

5.615.5.18 `template<typename _CharT, typename _Traits > basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::get (__streambuf_type & __sb, char_type __delim)`

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
<code>__delim</code>	A "stop" character.

Returns

*this

Characters are extracted and inserted into `__sb` until one of the following happens:

- the input sequence reaches EOF
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted)
- the next character equals `__delim` (in this case, the character is not extracted)
- an exception occurs (and in this case is caught)

If no characters are stored, failbit is set in the stream's error state.

Definition at line 356 of file istream.tcc.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_ios< _CharT, _Traits >::setstate()`, `std::basic_streambuf< _CharT, _Traits >::sgetc()`, `std::basic_streambuf< _CharT, _Traits >::snextc()`, and `std::basic_streambuf< _CharT, _Traits >::putc()`.

5.615.5.19 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits >::get (__streambuf_type & __sb) [inline]`

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
-------------------	-------------------------------------

Returns

*this

Returns `get(__sb,widen("\n"))`.

Definition at line 387 of file istream.

5.615.5.20 `template<typename _CharT, typename _Traits > basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::getline (char_type * __s, streamsize __n, char_type __delim)`

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.
<code>__delim</code>	A "stop" character.

Returns

*this

Extracts and stores characters into `__s` until one of the following happens. Note that these criteria are required to be tested in the order listed here, to allow an input line to exactly fill the `__s` array without setting failbit.

1. the input sequence reaches end-of-file, in which case eofbit is set in the stream error state
2. the next character equals `__delim`, in which case the character is extracted (and therefore counted in `gcount()`) but not stored
3. `__n-1` characters are stored, in which case failbit is set in the stream error state

If no characters are extracted, failbit is set. (An empty line of input should therefore not cause failbit to be set.)

In any case, a null character is stored in the next location in the array.

Definition at line 400 of file istream.tcc.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_streambuf< _CharT, _Traits >::sbumpc()`, `std::basic_ios< _CharT, _Traits >::setstate()`, `std::basic_streambuf< _CharT, _Traits >::sgetc()`, and `std::basic_streambuf< _CharT, _Traits >::snextc()`.

Referenced by `std::basic_istream< _CharT, _Traits >::get()`.

5.615.5.21 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::getline (char_type * __s, streamsize __n) [inline]`

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.

Returns

*this

Returns `getline(__s, __n, widen("\n"))`.

Definition at line 427 of file `istream`.

5.615.5.22 `template<> basic_istream<char> & std::basic_istream<char>::getline (char_type * __s, streamsize __n, char_type __delim)`

Explicit specialization declarations, defined in `src/istream.cc`.

5.615.5.23 `locale std::ios_base::getloc () const [inline], [inherited]`

Locale access.

Returns

A copy of the current locale.

If `imbue(loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale()`, the global C++ locale.

Definition at line 763 of file `ios_base.h`.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::money_put<_CharT, _Outiter>::do_put()`, `std::operator>>()`, and `std::ws()`.

5.615.5.24 `template<typename _CharT, typename _Traits> bool std::basic_ios<_CharT, _Traits>::good () const [inline], [inherited]`

Fast error checking.

Returns

True if no error flags are set.

A wrapper around `rdstate`.

Definition at line 180 of file `basic_ios.h`.

Referenced by `std::__detail::operator>>()`, `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`, and `std::basic_istream<_CharT, _Traits>::sentry::sentry()`.

5.615.5.25 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::ignore (streamsize __n, int_type __delim)`

Discarding characters.

Parameters

<code>__n</code>	Number of characters to discard.
<code>__delim</code>	A "stop" character.

Returns

*this

Extracts characters and throws them away until one of the following happens:

- if `__n != std::numeric_limits<int>::max()`, `__n` characters are extracted
- the input sequence reaches end-of-file
- the next character equals `__delim` (in this case, the character is extracted); note that this condition will never occur if `__delim` equals `traits::eof()`.

NB: Provide three overloads, instead of the single function (with defaults) mandated by the Standard: this leads to a better performing implementation, while still conforming to the Standard.

Definition at line 555 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_istream<_CharT, _Traits>::peek()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_streambuf<_CharT, _Traits>::sbumpc()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sgetc()`, and `std::basic_streambuf<_CharT, _Traits>::snextc()`.

5.615.5.26 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::ignore (streamsize __n)`

Simple extraction.

Returns

A character, or `eof()`.

Tries to extract a character. If none are available, sets failbit and returns `traits::eof()`.

Definition at line 493 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sgetc()`, and `std::basic_streambuf<_CharT, _Traits>::snextc()`.

5.615.5.27 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::ignore (void)`

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns traits::eof().

Definition at line 460 of file istream.tcc.

References std::basic_istream< _CharT, _Traits>::M_gcount, std::ios_base::badbit, std::ios_base::eofbit, std::ios_base::goodbit, std::basic_ios< _CharT, _Traits>::rdbuf(), std::basic_streambuf< _CharT, _Traits>::sbumpc(), and std::basic_ios< _CharT, _Traits>::setstate().

Referenced by std::basic_istream< _CharT, _Traits>::getline(), and std::basic_istream< _CharT, _Traits>::ignore().

5.615.5.28 `template<typename _CharT, typename _Traits> locale std::basic_ios< _CharT, _Traits>::imbue (const locale & __loc) [inherited]`

Moves to a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls `ios_base::imbue(loc)`, and if a stream buffer is associated with this stream, calls that buffer's `pubimbue(loc)`.

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Definition at line 114 of file basic_ios.tcc.

References std::ios_base::imbue().

Referenced by std::basic_ios< char, _Traits>::fill().

5.615.5.29 `template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits>::init (basic_streambuf< _CharT, _Traits> * __sb) [protected], [inherited]`

All setup is performed here.

This is called from the public constructor. It is not virtual and cannot be redefined.

Definition at line 126 of file basic_ios.tcc.

Referenced by std::basic_ios< char, _Traits>::basic_ios().

5.615.5.30 `long& std::ios_base::iword (int __ix) [inline],[inherited]`

Access to integer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to an integer associated with the index.

The `iword` function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 809 of file `ios_base.h`.

5.615.5.31 `template<typename _CharT, typename _Traits> char std::basic_ios<_CharT, _Traits>::narrow (char_type __c, char __dfault) const [inline],[inherited]`

Squeezes characters.

Parameters

<code>__c</code>	The character to narrow.
<code>__dfault</code>	The character to narrow.

Returns

The narrowed character.

Maps a character of `char_type` to a character of `char`, if possible.

Returns the result of

```
1 std::use_facet<ctype<char_type>> >(getloc()).narrow(c,dfault)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Definition at line 430 of file `basic_ios.h`.

5.615.5.32 `template<typename _CharT, typename _Traits> std::basic_ios<_CharT, _Traits>::operator bool () const`
`[inline], [explicit], [inherited]`

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`

Definition at line 117 of file `basic_ios.h`.

5.615.5.33 `template<typename _CharT, typename _Traits> bool std::basic_ios<_CharT, _Traits>::operator! () const`
`[inline], [inherited]`

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`

Definition at line 125 of file `basic_ios.h`.

5.615.5.34 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (__istream_type &(*)(__istream_type &)__pf)` `[inline]`

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

Definition at line 120 of file `istream`.

Referenced by `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::basic_istream<_CharT, _Traits>::sentry←::sentry()`, and `std::ws()`.

5.615.5.35 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (__ios_type &(*)(__ios_type &)__pf)` `[inline]`

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

Definition at line 124 of file `istream`.

5.615.5.36 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (ios_base &(*)(ios_base &)__pf)` `[inline]`

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

Definition at line 131 of file `istream`.

5.615.5.37 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (bool &__n)` `[inline]`

Integer arithmetic extractors.

Parameters

<code>__↔</code> <code>__n</code>	A variable of builtin integral type.
--------------------------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 168 of file `istream`.

5.615.5.38 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::operator>> (short & __n)`

Integer arithmetic extractors.

Parameters

<code>__↔</code> <code>__n</code>	A variable of builtin integral type.
--------------------------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 114 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::failbit`, `std::num_get< _CharT, _InIter>::get()`, `std::ios_base::goodbit`, `std::basic_istream< _CharT, _Traits>::operator>>()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

5.615.5.39 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::operator>> (unsigned short & __n) [inline]`

Integer arithmetic extractors.

Parameters

<code>__↔</code> <code>__n</code>	A variable of builtin integral type.
--------------------------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 175 of file `istream`.

5.615.5.40 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::operator>> (int & __n)`

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 159 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::failbit`, `std::num_get<_CharT, _InIter>::get()`, `std::ios_base::goodbit`, `std::basic_istream<_CharT, _Traits>::operator>>()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

5.615.5.41 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (unsigned int & __n) [inline]`

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 182 of file `istream`.

5.615.5.42 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::operator>> (long & __n) [inline]`

Integer arithmetic extractors.

Parameters

<code>_↔</code>	A variable of builtin integral type.
<code>_n</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 186 of file `istream`.

```
5.615.5.43 template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> ( unsigned long &_n ) [inline]
```

Integer arithmetic extractors.

Parameters

<code>_↔</code>	A variable of builtin integral type.
<code>_n</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 190 of file `istream`.

```
5.615.5.44 template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> ( long long &_n ) [inline]
```

Integer arithmetic extractors.

Parameters

<code>_↔</code>	A variable of builtin integral type.
<code>_n</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 195 of file `istream`.

```
5.615.5.45  template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits
              >::operator>> ( unsigned long long & __n ) [inline]
```

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 199 of file `istream`.

```
5.615.5.46  template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits
              >::operator>> ( float & __f ) [inline]
```

Floating point arithmetic extractors.

Parameters

<code>__f</code>	A variable of builtin floating point type.
------------------	--

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 214 of file `istream`.

```
5.615.5.47  template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits
              >::operator>> ( double & __f ) [inline]
```

Floating point arithmetic extractors.

Parameters

\leftrightarrow	A variable of builtin floating point type.
$_ \leftrightarrow$	
\leftrightarrow	
$_ \leftrightarrow$	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 218 of file `istream`.

5.615.5.48 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (long double &_f) [inline]`

Floating point arithmetic extractors.

Parameters

\leftrightarrow	A variable of builtin floating point type.
$_ \leftrightarrow$	
\leftrightarrow	
$_ \leftrightarrow$	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 222 of file `istream`.

5.615.5.49 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (void *&_p) [inline]`

Basic arithmetic extractors.

Parameters

$_ \leftrightarrow$	A variable of pointer type.
$_p$	

Returns

*`this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 235 of file `istream`.

5.615.5.50 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::operator>> (__streambuf_type * __sb)`

Extracting into another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is `NULL`, the stream will set `failbit` in its error state.

Characters are extracted from this stream and inserted into the `__sb` streambuf until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs (and in this case is caught)

If the function inserts no characters, `failbit` is set.

Definition at line 204 of file `istream.tcc`.

References `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::basic_istream< _CharT, _Traits>::get()`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

5.615.5.51 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits>::int_type std::basic_istream< _CharT, _Traits>::peek (void)`

Looking ahead in the stream.

Returns

The next character, or `eof()`.

If, after constructing the sentry object, `good()` is `false`, returns `traits::eof()`. Otherwise reads but does not extract the next input character.

Definition at line 620 of file `istream.tcc`.

References `std::basic_istream< _CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, `std::basic_istream< _CharT, _Traits>::read()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream< _CharT, _Traits>::ignore()`.

5.615.5.52 **streamsize** std::ios_base::precision () const [inline],[inherited]

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Definition at line 689 of file ios_base.h.

Referenced by std::basic_ios<_CharT, _Traits>::copyfmt(), std::num_get<_CharT, _InIter>::do_get(), std::normal_distribution<_RealType>::operator>(), std::gamma_distribution<_RealType>::operator>(), std::negative_binomial_distribution<_IntType>::operator>(), and std::operator>>().

5.615.5.53 **streamsize** std::ios_base::precision (streamsize __prec) [inline],[inherited]

Changing flags.

Parameters

<code>__prec</code>	The new precision value.
---------------------	--------------------------

Returns

The previous value of precision().

Definition at line 698 of file ios_base.h.

5.615.5.54 **template**<typename _CharT, typename _Traits> **basic_istream**<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::putback (char_type __c)

Unextracting a single character.

Parameters

<code>__c</code>	The character to push back into the input stream.
------------------	---

Returns

*this

If rdbuf () is not null, calls rdbuf ()->sputbackc (c) .

If rdbuf () is null or if sputbackc () fails, sets badbit in the error state.

Note

This function first clears eofbit. Since no characters are extracted, the next call to `gcount()` will return 0, as required by DR 60.

Definition at line 711 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::rdstate()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sputbackc()`, and `std::basic_istream<_CharT, _Traits>::unget()`.

Referenced by `std::basic_istream<_CharT, _Traits>::readsome()`.

5.615.5.55 `void*& std::ios_base::pword (int __ix)` `[inline]`, `[inherited]`

Access to void pointer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to a `void*` associated with the index.

The `pword` function provides access to an array of pointers that can be used for any purpose. The array grows as required to hold the supplied index. All pointers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 830 of file `ios_base.h`.

5.615.5.56 `template<typename _CharT, typename _Traits> basic_streambuf<_CharT, _Traits>* std::basic_ios<_CharT, _Traits>::rdbuf () const` `[inline]`, `[inherited]`

Accessing the underlying buffer.

Returns

The current stream buffer.

This does not change the state of the stream.

Definition at line 321 of file `basic_ios.h`.

Referenced by `std::basic_ostream<_CharT, _Traits>::flush()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_ostream<_CharT, _Traits>::operator<<()`, `std::tr2::operator>>()`, `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::operator>>()`, `std::basic_istream<_CharT, _Traits>::peek()`, `std::basic_ostream<_CharT, _Traits>::put()`, `std::basic_istream<_CharT, _Traits>::putback()`, `std::basic_ios<char, _Traits>::rdbuf()`, `std::basic_istream<_CharT, _Traits>::read()`, `std::basic_istream<_CharT, _Traits>::readsome()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_ostream<_CharT, _Traits>::seekp()`, `std::basic_istream<_CharT, _Traits>::sentry()`, `std::sentry()`, `std::basic_istream<_CharT, _Traits>::sync()`, `std::basic_istream<_CharT, _Traits>::tellg()`, `std::basic_ostream<_CharT, _Traits>::tellp()`, `std::basic_istream<_CharT, _Traits>::unget()`, and `std::ws()`.

5.615.5.57 `template<typename _CharT, typename _Traits> basic_streambuf< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::rdbuf (basic_streambuf< _CharT, _Traits > * __sb)` [inherited]

Changing the underlying buffer.

Parameters

<code>__sb</code>	The new stream buffer.
-------------------	------------------------

Returns

The previous stream buffer.

Associates a new buffer with the current stream, and clears the error state.

Due to historical accidents which the LWG refuses to correct, the I/O library suffers from a design error: this function is hidden in derived classes by overrides of the zero-argument `rdbuf()`, which is non-virtual for hysterical raisins. As a result, you must use explicit qualifications to access this function via any derived class. For example:

```
1 std::fstream    foo;           // or some other derived type
2 std::streambuf* p = .....;
3
4 foo.ios::rdbuf(p);           // ios == basic_ios<char>
```

Definition at line 53 of file `basic_ios.tcc`.

5.615.5.58 `template<typename _CharT, typename _Traits> iosstate std::basic_ios< _CharT, _Traits >::rdstate () const` [inline], [inherited]

Returns the error state of the stream buffer.

Returns

A bit pattern (well, isn't everything?)

See `std::ios_base::iosstate` for the possible bit values. Most users will call one of the interpreting wrappers, e.g., `good()`.

Definition at line 137 of file `basic_ios.h`.

Referenced by `std::basic_ios< char, _Traits >::bad()`, `std::basic_ios< char, _Traits >::eof()`, `std::basic_ios< char, _Traits >::fail()`, `std::basic_ios< char, _Traits >::good()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ios< char, _Traits >::setstate()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

5.615.5.59 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::read (char_type * __s, streamsize __n)`

Extraction without delimiters.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

`*this`

If the stream state is `good()`, extracts characters and stores them into `__s` until one of the following happens:

- `__n` characters are stored
- the input sequence reaches end-of-file, in which case the error state is set to `failbit|eofbit`.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 650 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_istream<_CharT, _Traits>::readsome()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream<_CharT, _Traits>::peek()`.

5.615.5.60 `template<typename _CharT, typename _Traits> streamsize std::basic_istream<_CharT, _Traits>::readsome (char_type * __s, streamsize __n)`

Extraction until the buffer is exhausted, but no more.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

The number of characters extracted.

Extracts characters and stores them into `__s` depending on the number of characters remaining in the streambuf's buffer, `rdbuf()->in_avail()`, called A here:

- if $A == -1$, sets eofbit and extracts no characters
- if $A == 0$, extracts no characters
- if $A > 0$, extracts $\min(A, n)$

The goal is to empty the current buffer, and to not request any more from the external input sequence controlled by the streambuf.

Definition at line 679 of file istream.tcc.

References std::basic_istream< _CharT, _Traits >::M_gcount, std::ios_base::badbit, std::ios_base::eofbit, std::ios_base::goodbit, std::min(), std::basic_istream< _CharT, _Traits >::putback(), std::basic_ios< _CharT, _Traits >::rdbuf(), and std::basic_ios< _CharT, _Traits >::setstate().

Referenced by std::basic_istream< _CharT, _Traits >::read().

5.615.5.61 void std::ios_base::register_callback (event_callback __fn, int __index) [inherited]

Add the callback __fn with parameter __index.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

5.615.5.62 template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::seekg (pos_type __pos)

Changing the current read position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

*this

If fail() is not true, calls rdbuf()->pubseekpos(__pos). If that function fails, sets failbit.

Note

This function first clears eofbit. It does not count the number of characters extracted, if any, and therefore does not affect the next call to gcount().

Definition at line 845 of file istream.tcc.

References `std::ios_base::badbit`, `std::basic_ios< _CharT, _Traits >::clear()`, `std::ios_base::eofbit`, `std::basic_ios< ↵
_CharT, _Traits >::fail()`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::ios_base::in`, `std::basic_ios< _CharT, _Traits
>::rdbuf()`, `std::basic_ios< _CharT, _Traits >::rdstate()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

Referenced by `std::basic_istream< _CharT, _Traits >::tellg()`.

5.615.5.63 `template<typename _CharT, typename _Traits > basic_istream< _CharT, _Traits > & std::basic_istream<
_CharT, _Traits >::seekg (off_type __off, ios_base::seekdir __dir)`

Changing the current read position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

`*this`

If `fail()` is not true, calls `rdbuf() -> pubseekoff(__off, __dir)`. If that function fails, sets failbit.

Note

This function first clears eofbit. It does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

Definition at line 884 of file istream.tcc.

References `std::ios_base::badbit`, `std::basic_ios< _CharT, _Traits >::clear()`, `std::ios_base::eofbit`, `std::basic_ios< ↵
_CharT, _Traits >::fail()`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::ios_base::in`, `std::basic_ios< _CharT, _Traits
>::rdbuf()`, `std::basic_ios< _CharT, _Traits >::rdstate()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

5.615.5.64 `fmtflags std::ios_base::setf (fmtflags __fmtfl) [inline], [inherited]`

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
----------------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set.

Definition at line 646 of file ios_base.h.

Referenced by std::boolalpha(), std::dec(), std::fixed(), std::hex(), std::hexfloat(), std::left(), std::oct(), std::__detail::operator>>(), std::right(), std::scientific(), std::showbase(), std::showpoint(), std::showpos(), std::skipws(), std::unitbuf(), and std::uppercase().

5.615.5.65 `fmtflags std::ios_base::setf (fmtflags __fmtfl, fmtflags __mask)` `[inline], [inherited]`

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
<code>__mask</code>	The flags mask for <i>fmtfl</i> .

Returns

The previous format control flags.

This function clears *mask* in the format flags, then sets *fmtfl* & *mask*. An example mask is `ios_base::adjustfield`.

Definition at line 663 of file ios_base.h.

5.615.5.66 `template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits >::setstate (iostate __state)` `[inline], [inherited]`

Sets additional flags in the error state.

Parameters

<code>__state</code>	The additional state flag(s) to set.
----------------------	--------------------------------------

See `std::ios_base::iostate` for the possible bit values.

Definition at line 157 of file basic_ios.h.

Referenced by std::basic_ostream< _CharT, _Traits >::flush(), std::basic_istream< _CharT, _Traits >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::getline(), std::basic_istream< _CharT, _Traits >::ignore(), std::basic_ostream< _CharT, _Traits >::operator<<(), std::operator<<(), std::basic_istream< _CharT, _Traits >::operator>>(), std::tr2::operator>>(), std::operator>>(), std::basic_istream< _CharT, _Traits >::peek(), std::basic_ostream< _CharT, _Traits >::put(), std::basic_istream< _CharT, _Traits >::putback(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::readsome(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_ostream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::sync(), std::basic_istream< _CharT, _Traits >::unget(), and std::ws().

5.615.5.67 `template<typename _CharT, typename _Traits> int std::basic_istream<_CharT, _Traits>::sync(void)`

Synchronizing the stream buffer.

Returns

0 on success, -1 on failure

If `rdbuf()` is a null pointer, returns -1.

Otherwise, calls `rdbuf()->pubsync()`, and if that returns -1, sets `badbit` and returns -1.

Otherwise, returns 0.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

Definition at line 781 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::goodbit`, `std::basic_streambuf<_CharT, _Traits>::pubsync()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::setstate()`, and `std::basic_istream<_CharT, _Traits>::tellg()`.

Referenced by `std::basic_istream<_CharT, _Traits>::unget()`.

5.615.5.68 `static bool std::ios_base::sync_with_stdio(bool __sync = true)` [static], [inherited]

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., `stdout`) and the standard C++ objects (e.g., `cout`). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstream.html#std.io.filestreams.binary>

5.615.5.69 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits>::pos_type std::basic_istream<_CharT, _Traits>::tellg(void)`

Getting the current read position.

Returns

A file position object.

If `fail()` is not false, returns `pos_type(-1)` to indicate failure. Otherwise returns `rdbuf() -> pubseekoff(0, cur, in)`.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`. At variance with `putback`, `unget` and `seekg`, `eofbit` is not cleared first.

Definition at line 817 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::cur`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::ios_base::in`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, and `std::basic_istream< _CharT, _Traits >::seekg()`.

Referenced by `std::basic_istream< _CharT, _Traits >::sync()`.

5.615.5.70 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits>* std::basic_ios< _CharT, _Traits>::tie() const` `[inline]`, `[inherited]`

Fetches the current *tied* stream.

Returns

A pointer to the tied stream, or NULL if the stream is not tied.

A stream may be *tied* (or synchronized) to a second output stream. When this stream performs any I/O, the tied stream is first flushed. For example, `std::cin` is tied to `std::cout`.

Definition at line 295 of file `basic_ios.h`.

Referenced by `std::basic_ios< char, _Traits >::basic_ios()`, `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_istream< _CharT, _Traits >::sentry::sentry()`.

5.615.5.71 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits>* std::basic_ios< _CharT, _Traits>::tie(basic_ostream< _CharT, _Traits>* __tiestr)` `[inline]`, `[inherited]`

Ties this stream to an output stream.

Parameters

<code>__tiestr</code>	The output stream.
-----------------------	--------------------

Returns

The previously tied output stream, or NULL if the stream was not tied.

This sets up a new tie; see `tie()` for more.

Definition at line 307 of file `basic_ios.h`.

5.615.5.72 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::unget (void)`

Unextracting the previous character.

Returns

`*this`

If `rdbuf()` is not null, calls `rdbuf()->sungetc(c)`.

If `rdbuf()` is null or if `sungetc()` fails, sets `badbit` in the error state.

Note

This function first clears `eofbit`. Since no characters are extracted, the next call to `gcount()` will return 0, as required by DR 60.

Definition at line 746 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::rdstate()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sungetc()`, and `std::basic_istream<_CharT, _Traits>::sync()`.

Referenced by `std::__detail::operator>>()`, and `std::basic_istream<_CharT, _Traits>::putback()`.

5.615.5.73 `void std::ios_base::unsetf (fmtflags __mask) [inline], [inherited]`

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

Definition at line 678 of file `ios_base.h`.

Referenced by `std::defaultfloat()`, `std::noboolalpha()`, `std::noshowbase()`, `std::noshowpoint()`, `std::noshowpos()`, `std::noskipws()`, `std::nounitbuf()`, and `std::nouppercase()`.

5.615.5.74 `template<typename _CharT, typename _Traits> char_type std::basic_ios<_CharT, _Traits>::widen (char __c) const [inline], [inherited]`

Widens characters.

Parameters

<code>_↔</code>	The character to widen.
<code>_C</code>	

Returns

The widened character.

Maps a character of `char` to a character of `char_type`.

Returns the result of

```
1 std::use_facet<ctype<char_type> >(getloc()).widen(c)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.↔html>

Definition at line 449 of file `basic_ios.h`.

Referenced by `std::basic_ios< char, _Traits >::fill()`, `std::getline()`, `std::normal_distribution< _RealType >::operator()()`, `std::gamma_distribution< _RealType >::operator()()`, `std::negative_binomial_distribution< _IntType >::operator()()`, `std::tr2::operator>>()`, and `std::operator>>()`.

5.615.5.75 `streamsize std::ios_base::width () const` `[inline]`, `[inherited]`

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Definition at line 712 of file `ios_base.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::money_get< _CharT, _Inlter >::do_get()`, `std::num_↔_get< _CharT, _Inlter >::do_get()`, `std::num_put< _CharT, _Outlter >::do_put()`, and `std::operator>>()`.

5.615.5.76 `streamsize std::ios_base::width (streamsize __wide)` `[inline]`, `[inherited]`

Changing flags.

Parameters

<code>__wide</code>	The new width value.
---------------------	----------------------

Returns

The previous value of `width()`.

Definition at line 721 of file `ios_base.h`.

5.615.5.77 `static int std::ios_base::xalloc () throw` `[static], [inherited]`

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the `iword` and `pword` functions. The expectation is that an application calls `xalloc` in order to obtain an index in the `iword` and `pword` arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. `xalloc` is guaranteed to return an index that is safe to use in the `iword` and `pword` arrays.

5.615.6 Member Data Documentation

5.615.6.1 `template<typename _CharT, typename _Traits> streamsize std::basic_istream< _CharT, _Traits >::M_gcount`
`[protected]`

The number of characters extracted in the previous unformatted function; see `gcount()`.

Definition at line 82 of file `istream`.

Referenced by `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

5.615.6.2 `const fmtflags std::ios_base::adjustfield` `[static], [inherited]`

A mask of `left|right|internal`. Useful for the 2-arg form of `setf`.

Definition at line 378 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::internal()`, `std::left()`, and `std::right()`.

5.615.6.3 `const openmode std::ios_base::app` `[static], [inherited]`

Seek to end before each write.

Definition at line 432 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

5.615.6.4 const openmode std::ios_base::ate [static],[inherited]

Open and seek to end immediately after opening.

Definition at line 435 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::open().

5.615.6.5 const iostate std::ios_base::badbit [static],[inherited]

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Definition at line 402 of file ios_base.h.

Referenced by std::basic_ios< char, _Traits >::bad(), std::basic_ios< char, _Traits >::fail(), std::basic_ostream< _CharT, _Traits >::flush(), std::basic_istream< _CharT, _Traits >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::basic_istream< _CharT, _Traits >::ignore(), std::basic_ostream< _CharT, _Traits >::operator<<(), std::operator<<(), std::basic_istream< _CharT, _Traits >::operator>>(), std::operator>>(), std::basic_istream< _CharT, _Traits >::peek(), std::basic_ostream< _CharT, _Traits >::put(), std::basic_istream< _CharT, _Traits >::putback(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::readsome(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_ostream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::sync(), std::basic_istream< _CharT, _Traits >::tellg(), std::basic_ostream< _CharT, _Traits >::tellp(), std::basic_istream< _CharT, _Traits >::unset(), and std::basic_ostream< _CharT, _Traits >::write().

5.615.6.6 const fmtflags std::ios_base::basefield [static],[inherited]

A mask of dec|oct|hex. Useful for the 2-arg form of setf.

Definition at line 381 of file ios_base.h.

Referenced by std::dec(), std::num_get< _CharT, _InIter >::do_get(), std::num_put< _CharT, _OutIter >::do_put(), std::hex(), std::oct(), and std::basic_ostream< _CharT, _Traits >::operator<<().

5.615.6.7 const seekdir std::ios_base::beg [static],[inherited]

Request a seek relative to the beginning of the stream.

Definition at line 464 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::seekpos(), and __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync().

5.615.6.8 const openmode std::ios_base::binary [static],[inherited]

Perform input and output in binary mode (as opposed to text mode). This is probably not what you think it is; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.←filestreams.binary>.

Definition at line 440 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::showmanyc().

5.615.6.9 `const fmtflags std::ios_base::boolalpha` `[static], [inherited]`

Insert/extract `bool` in alphabetic rather than numeric format.

Definition at line 326 of file `ios_base.h`.

Referenced by `std::boolalpha()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_put<_CharT, _OutIter>::do_put()`, and `std::noboolalpha()`.

5.615.6.10 `const seekdir std::ios_base::cur` `[static], [inherited]`

Request a seek relative to the current position within the sequence.

Definition at line 467 of file `ios_base.h`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::imbue()`, `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_filebuf<_CharT, _Traits>::pbackfail()`, `std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekoff()`, `std::basic_filebuf<_CharT, _Traits>::seekoff()`, `__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>::sync()`, `std::basic_istream<_CharT, _Traits>::tellg()`, and `std::basic_ostream<_CharT, _Traits>::tellp()`.

5.615.6.11 `const fmtflags std::ios_base::dec` `[static], [inherited]`

Converts integer input or generates integer output in decimal base.

Definition at line 329 of file `ios_base.h`.

Referenced by `std::dec()`.

5.615.6.12 `const seekdir std::ios_base::end` `[static], [inherited]`

Request a seek relative to the current end of the sequence.

Definition at line 470 of file `ios_base.h`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::open()`, and `std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekoff()`.

5.615.6.13 `const iostate std::ios_base::eofbit` `[static], [inherited]`

Indicates that an input operation reached the end of an input sequence.

Definition at line 405 of file `ios_base.h`.

Referenced by `std::time_get<_CharT, _InIter>::do_get()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::time_get<_CharT, _InIter>::do_get_date()`, `std::time_get<_CharT, _InIter>::do_get_monthname()`, `std::time_get<_CharT, _InIter>::do_get_time()`, `std::time_get<_CharT, _InIter>::do_get_weekday()`, `std::time_get<_CharT, _InIter>::do_get_year()`, `std::basic_ios<char, _Traits>::eof()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::time_get<_CharT, _InIter>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::operator>>()`, `std::basic_istream<_CharT, _Traits>::peek()`, `std::basic_istream<_CharT, _Traits>::putback()`, `std::basic_istream<_CharT, _Traits>::read()`, `std::basic_istream<_CharT, _Traits>::readsome()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_istream<_CharT, _Traits>::unget()`, and `std::ws()`.

5.615.6.14 const iostate std::ios_base::failbit [static], [inherited]

Indicates that an input operation failed to read the expected characters, or that an output operation failed to generate the desired characters.

Definition at line 410 of file ios_base.h.

Referenced by std::time_get< _CharT, _Inlter >::do_date_order(), std::num_get< _CharT, _Inlter >::do_get(), std::time_get< _CharT, _Inlter >::do_get_monthname(), std::time_get< _CharT, _Inlter >::do_get_weekday(), std::time_get< _CharT, _Inlter >::do_get_year(), std::basic_ios< char, _Traits >::fail(), std::basic_istream< _CharT, _Traits >::get(), std::time_get< _CharT, _Inlter >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::basic_ostream< _CharT, _Traits >::operator<<(), std::basic_istream< _CharT, _Traits >::operator>>(), std::operator>>(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_ostream< _CharT, _Traits >::sentry::sentry(), and std::basic_istream< _CharT, _Traits >::sentry::sentry().

5.615.6.15 const fmtflags std::ios_base::fixed [static], [inherited]

Generate floating-point output in fixed-point notation.

Definition at line 332 of file ios_base.h.

Referenced by std::num_get< _CharT, _Inlter >::do_get(), std::fixed(), and std::hexfloat().

5.615.6.16 const fmtflags std::ios_base::floatfield [static], [inherited]

A mask of scientific|fixed. Useful for the 2-arg form of setf.

Definition at line 384 of file ios_base.h.

Referenced by std::defaultfloat(), std::num_get< _CharT, _Inlter >::do_get(), std::fixed(), std::hexfloat(), and std::scientific().

5.615.6.17 const iostate std::ios_base::goodbit [static], [inherited]

Indicates all is well.

Definition at line 413 of file ios_base.h.

Referenced by std::time_get< _CharT, _Inlter >::do_date_order(), std::time_get< _CharT, _Inlter >::do_get(), std::num_get< _CharT, _Inlter >::do_get(), std::time_get< _CharT, _Inlter >::do_get_monthname(), std::time_get< _CharT, _Inlter >::do_get_weekday(), std::time_get< _CharT, _Inlter >::do_get_year(), std::basic_ostream< _CharT, _Traits >::flush(), std::basic_istream< _CharT, _Traits >::get(), std::time_get< _CharT, _Inlter >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::basic_istream< _CharT, _Traits >::ignore(), std::basic_ostream< _CharT, _Traits >::operator<<(), std::basic_istream< _CharT, _Traits >::operator>>(), std::operator>>(), std::basic_istream< _CharT, _Traits >::peek(), std::basic_ostream< _CharT, _Traits >::put(), std::basic_istream< _CharT, _Traits >::putback(), std::basic_ios< char, _Traits >::rdstate(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::readsome(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_ostream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::sync(), and std::basic_istream< _CharT, _Traits >::unset().

5.615.6.18 const fmtflags std::ios_base::hex [static],[inherited]

Converts integer input or generates integer output in hexadecimal base.

Definition at line 335 of file ios_base.h.

Referenced by std::num_get< _CharT, _Inlter >::do_get(), std::num_put< _CharT, _Outlter >::do_put(), std::hex(), and std::basic_ostream< _CharT, _Traits >::operator<<().

5.615.6.19 const openmode std::ios_base::in [static],[inherited]

Open for input. Default for ifstream and fstream.

Definition at line 443 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::pbackfail(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos(), std::basic_filebuf< _CharT, _Traits >::showmanyc(), __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync(), std::basic_istream< _CharT, _Traits >::tellg(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::underflow(), std::basic_filebuf< _CharT, _Traits >::underflow(), and std::basic_filebuf< _CharT, _Traits >::xsgetn().

5.615.6.20 const fmtflags std::ios_base::internal [static],[inherited]

Adds fill characters at a designated internal point in certain generated output, or identical to right if no such point is designated.

Definition at line 340 of file ios_base.h.

Referenced by std::money_get< _CharT, _Inlter >::do_get(), std::num_put< _CharT, _Outlter >::do_put(), and std::internal().

5.615.6.21 const fmtflags std::ios_base::left [static],[inherited]

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.)

Definition at line 344 of file ios_base.h.

Referenced by std::money_get< _CharT, _Inlter >::do_get(), std::num_put< _CharT, _Outlter >::do_put(), and std::left().

5.615.6.22 const fmtflags std::ios_base::oct [static],[inherited]

Converts integer input or generates integer output in octal base.

Definition at line 347 of file ios_base.h.

Referenced by std::num_get< _CharT, _Inlter >::do_get(), std::oct(), and std::basic_ostream< _CharT, _Traits >::operator<<().

5.615.6.23 const openmode std::ios_base::out [static],[inherited]

Open for output. Default for ofstream and fstream.

Definition at line 446 of file ios_base.h.

Referenced by std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow(), std::basic_filebuf< _CharT, _Traits >::overflow(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::pbackfail(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos(), __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync(), std::basic_ostream< _CharT, _Traits >::tellp(), and std::basic_filebuf< _CharT, _Traits >::xsputn().

5.615.6.24 const fmtflags std::ios_base::right [static],[inherited]

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Definition at line 351 of file ios_base.h.

Referenced by std::right().

5.615.6.25 const fmtflags std::ios_base::scientific [static],[inherited]

Generates floating-point output in scientific notation.

Definition at line 354 of file ios_base.h.

Referenced by std::hexfloat(), and std::scientific().

5.615.6.26 const fmtflags std::ios_base::showbase [static],[inherited]

Generates a prefix indicating the numeric base of generated integer output.

Definition at line 358 of file ios_base.h.

Referenced by std::money_get< _CharT, _InIter >::do_get(), std::num_get< _CharT, _InIter >::do_get(), std::num_put< _CharT, _OutIter >::do_put(), std::noshowbase(), and std::showbase().

5.615.6.27 const fmtflags std::ios_base::showpoint [static],[inherited]

Generates a decimal-point character unconditionally in generated floating-point output.

Definition at line 362 of file ios_base.h.

Referenced by std::noshowpoint(), and std::showpoint().

5.615.6.28 const fmtflags std::ios_base::showpos [static],[inherited]

Generates a + sign in non-negative generated numeric output.

Definition at line 365 of file ios_base.h.

Referenced by std::num_get< _CharT, _InIter >::do_get(), std::noshowpos(), and std::showpos().

5.615.6.29 `const fmtflags std::ios_base::skipws` `[static], [inherited]`

Skips leading white space before certain input operations.

Definition at line 368 of file `ios_base.h`.

Referenced by `std::noskipws()`, `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, and `std::skipws()`.

5.615.6.30 `const openmode std::ios_base::trunc` `[static], [inherited]`

Open for input. Default for `ofstream`.

Definition at line 449 of file `ios_base.h`.

5.615.6.31 `const fmtflags std::ios_base::unitbuf` `[static], [inherited]`

Flushes output after each output operation.

Definition at line 371 of file `ios_base.h`.

Referenced by `std::nunitbuf()`, and `std::unitbuf()`.

5.615.6.32 `const fmtflags std::ios_base::uppercase` `[static], [inherited]`

Replaces certain lowercase letters with their uppercase equivalents in generated output.

Definition at line 375 of file `ios_base.h`.

Referenced by `std::num_get<_CharT, _Intler>::do_get()`, `std::num_put<_CharT, _Outlter>::do_put()`, `std::noupเปอร์case()`, and `std::uppercase()`.

The documentation for this class was generated from the following files:

- [iosfwd](#)
- [istream](#)
- [istream.tcc](#)

5.616 `std::basic_istream<_CharT, _Traits>::sentry` Class Reference

Public Types

- typedef `__istream_type::__ctype_type` `__ctype_type`
- typedef `_Traits::int_type` `__int_type`
- typedef `basic_istream<_CharT, _Traits>` `__istream_type`
- typedef `basic_streambuf<_CharT, _Traits>` `__streambuf_type`
- typedef `_Traits` `traits_type`

Public Member Functions

- [sentry](#) ([basic_istream](#)< _CharT, _Traits > &__is, bool __noskipws=false)
- [operator bool](#) () const

5.616.1 Detailed Description

```
template<typename _CharT, typename _Traits>
class std::basic_istream< _CharT, _Traits >::sentry
```

Performs setup work for input streams.

Objects of this class are created before all of the standard extractors are run. It is responsible for *exception-safe prefix and suffix operations*, although only prefix actions are currently required by the standard.

Definition at line 686 of file istream.

5.616.2 Member Typedef Documentation

```
5.616.2.1 template<typename _CharT, typename _Traits> typedef _Traits std::basic_istream< _CharT, _Traits
>::sentry::traits_type
```

Easy access to dependent types.

Definition at line 693 of file istream.

5.616.3 Constructor & Destructor Documentation

```
5.616.3.1 template<typename _CharT, typename _Traits> std::basic_istream< _CharT, _Traits >::sentry::sentry (
    basic_istream< _CharT, _Traits > & __is, bool __noskipws = false ) [explicit]
```

The constructor performs all the work.

Parameters

<code>__is</code>	The input stream to guard.
<code>__noskipws</code>	Whether to consume whitespace or not.

If the stream state is good (`__is.good()` is true), then the following actions are performed, otherwise the sentry state is false (*not okay*) and failbit is set in the stream state.

The sentry's preparatory actions are:

1. if the stream is tied to an output stream, `is.tie()->flush()` is called to synchronize the output sequence

2. if `__noskipws` is false, and `ios_base::skipws` is set in `is.flags()`, the sentry extracts and discards whitespace characters from the stream. The currently imbued locale is used to determine whether each character is whitespace.

If the stream state is still good, then the sentry state becomes true (*okay*).

Definition at line 47 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::flags()`, `std::num_get<_CharT, _InIter>::get()`, `std::basic_ios<_CharT, _Traits>::good()`, `std::ios_base::goodbit`, `std::__ctype_abstract_base<_CharT>::is()`, `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sgetc()`, `std::ios_base::skipws`, `std::basic_streambuf<_CharT, _Traits>::snextc()`, and `std::basic_ios<_CharT, _Traits>::tie()`.

5.616.4 Member Function Documentation

5.616.4.1 `template<typename _CharT, typename _Traits> std::basic_istream<_CharT, _Traits>::sentry::operator bool ()`
`const [inline], [explicit]`

Quick status checking.

Returns

The sentry state.

For ease of use, sentries may be converted to booleans. The return value is that of the sentry state (`true == okay`).

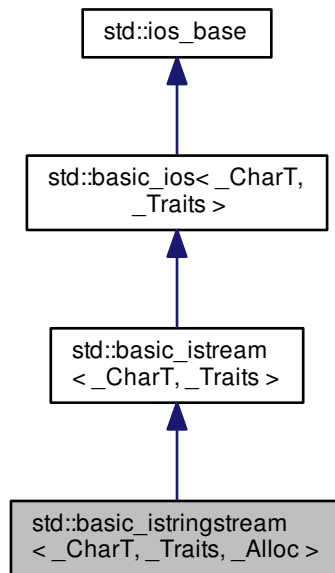
Definition at line 734 of file `istream`.

The documentation for this class was generated from the following files:

- [istream](#)
- [istream.tcc](#)

5.617 std::basic_istream< _CharT, _Traits, _Alloc > Class Template Reference

Inheritance diagram for std::basic_istream< _CharT, _Traits, _Alloc >:



Public Types

- typedef `ctype< _CharT > __ctype_type`
- typedef `basic_ios< _CharT, _Traits > __ios_type`
- typedef `basic_istream< char_type, traits_type > __istream_type`
- typedef `num_get< _CharT, istreambuf_iterator< _CharT, _Traits > > __num_get_type`
- typedef `basic_streambuf< _CharT, _Traits > __streambuf_type`
- typedef `basic_string< _CharT, _Traits, _Alloc > __string_type`
- typedef `basic_stringbuf< _CharT, _Traits, _Alloc > __stringbuf_type`
- typedef `_Alloc allocator_type`
- typedef `_CharT char_type`
- enum `event { erase_event, imbue_event, copyfmt_event }`
- typedef `void(* event_callback) (event __e, ios_base & __b, int __i)`
- typedef `_ios_Fmtflags fmtflags`
- typedef `traits_type::int_type int_type`
- typedef `int io_state`
- typedef `_ios_istate istate`
- typedef `traits_type::off_type off_type`
- typedef `int open_mode`
- typedef `_ios_Openmode openmode`
- typedef `traits_type::pos_type pos_type`

- typedef int **seek_dir**
- typedef _ios_Seekdir **seekdir**
- typedef **std::streamoff** **streamoff**
- typedef **std::streampos** **streampos**
- typedef _Traits **traits_type**
- typedef **num_put**< _CharT, **ostreambuf_iterator**< _CharT, _Traits > > **__num_put_type**

Public Member Functions

- **basic_istream** (**ios_base::openmode** __mode=**ios_base::in**)
- **basic_istream** (const __string_type &__str, **ios_base::openmode** __mode=**ios_base::in**)
- **basic_istream** (const **basic_istream** &)=delete
- **basic_istream** (**basic_istream** &&__rhs)
- ~**basic_istream** ()
- template<typename _ValueT >
basic_istream< _CharT, _Traits > & **M_extract** (_ValueT &__v)
- const **locale** & **M_getloc** () const
- void **M_setstate** (**iostate** __state)
- bool **bad** () const
- void **clear** (**iostate** __state=**goodbit**)
- **basic_ios** & **copyfmt** (const **basic_ios** &__rhs)
- bool **eof** () const
- **iostate** **exceptions** () const
- void **exceptions** (**iostate** __except)
- bool **fail** () const
- char_type **fill** () const
- char_type **fill** (char_type __ch)
- **fmtflags** **flags** () const
- **fmtflags** **flags** (**fmtflags** __fmtfl)
- **streamsize** **gcount** () const
- template<>
basic_istream< char > & **getline** (char_type * __s, **streamsize** __n, char_type __delim)
- template<>
basic_istream< wchar_t > & **getline** (char_type * __s, **streamsize** __n, char_type __delim)
- **locale** **getloc** () const
- bool **good** () const
- template<>
basic_istream< char > & **ignore** (**streamsize** __n)
- template<>
basic_istream< char > & **ignore** (**streamsize** __n, int_type __delim)
- template<>
basic_istream< wchar_t > & **ignore** (**streamsize** __n)
- template<>
basic_istream< wchar_t > & **ignore** (**streamsize** __n, int_type __delim)
- **locale** **imbue** (const **locale** &__loc)
- long & **inword** (int __ix)
- char **narrow** (char_type __c, char __dfault) const
- **basic_istream** & **operator=** (const **basic_istream** &)=delete
- **basic_istream** & **operator=** (**basic_istream** &&__rhs)

- [__istream_type & operator>>](#) (void *&__p)
 - [__istream_type & operator>>](#) ([__streambuf_type](#) * __sb)
 - [streamsize precision](#) () const
 - [streamsize precision](#) ([streamsize](#) __prec)
 - void *& [pword](#) (int __ix)
 - [basic_streambuf](#)< _CharT, _Traits > * [rdbuf](#) ([basic_streambuf](#)< _CharT, _Traits > * __sb)
 - [__stringbuf_type](#) * [rdbuf](#) () const
 - [iostate rdstate](#) () const
 - void [register_callback](#) ([event_callback](#) __fn, int __index)
 - [fmtflags setf](#) ([fmtflags](#) __fmtfl)
 - [fmtflags setf](#) ([fmtflags](#) __fmtfl, [fmtflags](#) __mask)
 - void [setstate](#) ([iostate](#) __state)
 - [__string_type](#) str () const
 - void str (const [__string_type](#) &__s)
 - void [swap](#) ([basic_istream](#) &__rhs)
 - [basic_ostream](#)< _CharT, _Traits > * [tie](#) () const
 - [basic_ostream](#)< _CharT, _Traits > * [tie](#) ([basic_ostream](#)< _CharT, _Traits > * __tiestr)
 - void [unsetf](#) ([fmtflags](#) __mask)
 - char_type [widen](#) (char __c) const
 - [streamsize width](#) () const
 - [streamsize width](#) ([streamsize](#) __wide)
-
- [__istream_type & operator>>](#) ([__istream_type](#) &(*__pf)([__istream_type](#) &))
 - [__istream_type & operator>>](#) ([__ios_type](#) &(*__pf)([__ios_type](#) &))
 - [__istream_type & operator>>](#) ([ios_base](#) &(*__pf)([ios_base](#) &))

Extractors

All the *operator>>* functions (aka formatted input functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (*noskipws*) set to false. This has several effects, concluding with the setting of a status flag; see the *sentry* documentation for more.

If the *sentry* status is good, the function tries to extract whatever data is appropriate for the type of the argument.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- [__istream_type & operator>>](#) (bool &__n)
 - [__istream_type & operator>>](#) (short &__n)
 - [__istream_type & operator>>](#) (unsigned short &__n)
 - [__istream_type & operator>>](#) (int &__n)
 - [__istream_type & operator>>](#) (unsigned int &__n)
 - [__istream_type & operator>>](#) (long &__n)
 - [__istream_type & operator>>](#) (unsigned long &__n)
 - [__istream_type & operator>>](#) (long long &__n)
 - [__istream_type & operator>>](#) (unsigned long long &__n)
-
- [__istream_type & operator>>](#) (float &__f)
 - [__istream_type & operator>>](#) (double &__f)
 - [__istream_type & operator>>](#) (long double &__f)

Unformatted Input Functions

All the unformatted input functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (`noskipws`) set to `true`. This has several effects, concluding with the setting of a status flag; see the `sentry` documentation for more.

If the `sentry` status is good, the function tries to extract whatever data is appropriate for the type of the argument.

The number of characters extracted is stored for later retrieval by `gcount()`.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- `int_type get ()`
 - `__istream_type & get (char_type &__c)`
 - `__istream_type & get (char_type * __s, streamsize __n, char_type __delim)`
 - `__istream_type & get (char_type * __s, streamsize __n)`
 - `__istream_type & get (__streambuf_type &__sb, char_type __delim)`
 - `__istream_type & get (__streambuf_type &__sb)`
 - `__istream_type & getline (char_type * __s, streamsize __n, char_type __delim)`
 - `__istream_type & getline (char_type * __s, streamsize __n)`
 - `__istream_type & ignore (streamsize __n, int_type __delim)`
 - `__istream_type & ignore (streamsize __n)`
 - `__istream_type & ignore ()`
 - `int_type peek ()`
 - `__istream_type & read (char_type * __s, streamsize __n)`
 - `streamsize readsome (char_type * __s, streamsize __n)`
 - `__istream_type & putback (char_type __c)`
 - `__istream_type & unget ()`
 - `int sync ()`
 - `pos_type tellg ()`
 - `__istream_type & seekg (pos_type)`
 - `__istream_type & seekg (off_type, ios_base::seekdir)`
-
- `operator bool () const`
 - `bool operator! () const`

Static Public Member Functions

- `static bool sync_with_stdio (bool __sync=true)`
- `static int xalloc () throw ()`

Static Public Attributes

- `static const fmtflags adjustfield`
- `static const openmode app`
- `static const openmode ate`
- `static const iostate badbit`
- `static const fmtflags basefield`
- `static const seekdir beg`
- `static const openmode binary`
- `static const fmtflags boolalpha`
- `static const seekdir cur`
- `static const fmtflags dec`
- `static const seekdir end`

- static const `iosate eofbit`
- static const `iosate failbit`
- static const `fmtflags fixed`
- static const `fmtflags floatfield`
- static const `iosate goodbit`
- static const `fmtflags hex`
- static const `openmode in`
- static const `fmtflags internal`
- static const `fmtflags left`
- static const `fmtflags oct`
- static const `openmode out`
- static const `fmtflags right`
- static const `fmtflags scientific`
- static const `fmtflags showbase`
- static const `fmtflags showpoint`
- static const `fmtflags showpos`
- static const `fmtflags skipws`
- static const `openmode trunc`
- static const `fmtflags unitbuf`
- static const `fmtflags uppercase`

Protected Types

- enum { `_S_local_word_size` }

Protected Member Functions

- void `_M_cache_locale` (const `locale` & __loc)
- void `_M_call_callbacks` (`event` __ev) throw ()
- void `_M_dispose_callbacks` (void) throw ()
- template<typename _ValueT >
`__istream_type` & `_M_extract` (_ValueT & __v)
- `_Words` & `_M_grow_words` (int __index, bool __iword)
- void `_M_init` () throw ()
- void `_M_move` (`ios_base` &) noexcept
- void `_M_swap` (`ios_base` & __rhs) noexcept
- void `init` (`basic_streambuf`<_CharT, _Traits> * __sb)
- void `move` (`basic_ios` & __rhs)
- void `move` (`basic_ios` && __rhs)
- void `set_rdbuf` (`basic_streambuf`<_CharT, _Traits> * __sb)
- void `swap` (`basic_ios` & __rhs) noexcept
- void `swap` (`basic_istream` & __rhs)

Protected Attributes

- `_Callback_list * _M_callbacks`
- `const __ctype_type * _M_ctype`
- `iosstate _M_exception`
- `char_type _M_fill`
- `bool _M_fill_init`
- `fmtflags _M_flags`
- `streamsize _M_gcount`
- `locale _M_ios_locale`
- `_Words _M_local_word [_S_local_word_size]`
- `const __num_get_type * _M_num_get`
- `const __num_put_type * _M_num_put`
- `streamsize _M_precision`
- `basic_streambuf<_CharT, _Traits> * _M_streambuf`
- `iosstate _M_streambuf_state`
- `basic_ostream<_CharT, _Traits> * _M_tie`
- `streamsize _M_width`
- `_Words * _M_word`
- `int _M_word_size`
- `_Words _M_word_zero`

5.617.1 Detailed Description

```
template<typename _CharT, typename _Traits, typename _Alloc>
class std::basic_istream<_CharT, _Traits, _Alloc>
```

Controlling input for `std::string`.

Template Parameters

<code>_CharT</code>	Type of character stream.
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_CharT></code> .

This class supports reading from objects of type `std::basic_string`, using the inherited functions from `std::basic_istream`. To control the associated sequence, an instance of `std::basic_stringbuf` is used, which this page refers to as `sb`.

Definition at line 100 of file `iosfwd`.

5.617.2 Member Typedef Documentation

```
5.617.2.1 template<typename _CharT, typename _Traits> typedef num_put<_CharT, ostreambuf_iterator<_CharT, _Traits>
> std::basic_ios<_CharT, _Traits>::__num_put_type [inherited]
```

These are non-standard types.

Definition at line 89 of file `basic_ios.h`.

5.617.2.2 `typedef void(* std::ios_base::event_callback)(event __e, ios_base &__b, int __i)` [inherited]

The type of an event callback function.

Parameters

<code>__e</code>	One of the members of the event enum.
<code>__b</code>	Reference to the <code>ios_base</code> object.
<code>__i</code>	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several `ios_base` and `basic_ios` functions, specifically `imbue()`, `copyfmt()`, and `~ios()`.

Definition at line 504 of file `ios_base.h`.

5.617.2.3 `typedef _ios_Fmtflags std::ios_base::fmtflags` [inherited]

This is a bitmask type.

`_Ios_Fmtflags` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `fmtflags` are:

- `boolalpha`
- `dec`
- `fixed`
- `hex`
- `internal`
- `left`
- `oct`
- `right`
- `scientific`
- `showbase`
- `showpoint`
- `showpos`
- `skipws`
- `unitbuf`
- `uppercase`
- `adjustfield`
- `basefield`
- `floatfield`

Definition at line 323 of file `ios_base.h`.

5.617.2.4 `typedef _Ios_Iostate std::ios_base::iostate` [inherited]

This is a bitmask type.

`_Ios_Iostate` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `iostate` are:

- `badbit`
- `eofbit`
- `failbit`
- `goodbit`

Definition at line 398 of file `ios_base.h`.

5.617.2.5 `typedef _Ios_Openmode std::ios_base::openmode` [inherited]

This is a bitmask type.

`_Ios_Openmode` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `openmode` are:

- `app`
- `ate`
- `binary`
- `in`
- `out`
- `trunc`

Definition at line 429 of file `ios_base.h`.

5.617.2.6 `typedef _Ios_Seekdir std::ios_base::seekdir` [inherited]

This is an enumerated type.

`_Ios_Seekdir` is implementation-defined. Defined values of type `seekdir` are:

- `beg`
- `cur`, equivalent to `SEEK_CUR` in the C standard library.
- `end`, equivalent to `SEEK_END` in the C standard library.

Definition at line 461 of file `ios_base.h`.

5.617.3 Member Enumeration Documentation

5.617.3.1 enum std::ios_base::event [inherited]

The set of events that may be passed to an event callback.

erase_event is used during ~ios() and copyfmt(). imbue_event is used during imbue(). copyfmt_event is used during copyfmt().

Definition at line 487 of file ios_base.h.

5.617.4 Constructor & Destructor Documentation

5.617.4.1 template<typename _CharT, typename _Traits, typename _Alloc> std::basic_istream<_CharT, _Traits, _Alloc>::basic_istream (ios_base::openmode __mode = ios_base::in) [inline], [explicit]

Default constructor starts with an empty string buffer.

Parameters

<code>__mode</code>	Whether the buffer can read, or write, or both.
---------------------	---

ios_base::in is automatically included in `__mode`.

Initializes `sb` using `__mode|in`, and passes `&sb` to the base class initializer. Does not allocate any buffer.

That's a lie. We initialize the base class with NULL, because the string class does its own memory management.

Definition at line 403 of file sstream.

5.617.4.2 template<typename _CharT, typename _Traits, typename _Alloc> std::basic_istream<_CharT, _Traits, _Alloc>::basic_istream (const __string_type & __str, ios_base::openmode __mode = ios_base::in) [inline], [explicit]

Starts with an existing string buffer.

Parameters

<code>__str</code>	A string to copy as a starting buffer.
<code>__mode</code>	Whether the buffer can read, or write, or both.

ios_base::in is automatically included in `mode`.

Initializes `sb` using `str` and `mode|in`, and passes `&sb` to the base class initializer.

That's a lie. We initialize the base class with NULL, because the string class does its own memory management.

Definition at line 421 of file sstream.

5.617.4.3 `template<typename _CharT, typename _Traits, typename _Alloc> std::basic_istream< _CharT, _Traits, _Alloc>::~~basic_istream () [inline]`

The destructor does nothing.

The buffer is deallocated by the stringbuf object, not the formatting stream.

Definition at line 432 of file sstream.

5.617.5 Member Function Documentation

5.617.5.1 `const locale& std::ios_base::M_getloc () const [inline],[inherited]`

Locale access.

Returns

A reference to the current locale.

Like getloc above, but returns a reference instead of generating a copy.

Definition at line 774 of file ios_base.h.

Referenced by `std::time_get< _CharT, _Inlter >::do_date_order()`, `std::time_get< _CharT, _Inlter >::do_get()`, `std::money_get< _CharT, _Inlter >::do_get()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::time_get< _CharT, _Inlter >::do_get_date()`, `std::time_get< _CharT, _Inlter >::do_get_monthname()`, `std::time_get< _CharT, _Inlter >::do_get_time()`, `std::time_get< _CharT, _Inlter >::do_get_weekday()`, `std::time_get< _CharT, _Inlter >::do_get_year()`, `std::time_put< _CharT, _Outlter >::do_put()`, `std::num_put< _CharT, _Outlter >::do_put()`, `std::time_get< _CharT, _Inlter >::get()`, and `std::time_put< _CharT, _Outlter >::put()`.

5.617.5.2 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::bad () const [inline],[inherited]`

Fast error checking.

Returns

True if the badbit is set.

Note that other iostate flags may also be set.

Definition at line 211 of file basic_ios.h.

5.617.5.3 `template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits >::clear (iostate __state = goodbit) [inherited]`

[Re]sets the error state.

Parameters

<code>__state</code>	The new state flag(s) to set.
----------------------	-------------------------------

See `std::ios_base::iostate` for the possible bit values. Most users will not need to pass an argument.

Definition at line 41 of file `basic_ios.tcc`.

Referenced by `std::basic_ios< char, _Traits >::exceptions()`, `std::__detail::operator>>()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_ios< char, _Traits >::rdstate()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ios< char, _Traits >::setstate()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

5.617.5.4 `template<typename _CharT, typename _Traits> basic_ios< _CharT, _Traits> & std::basic_ios< _CharT, _Traits>::copyfmt (const basic_ios< _CharT, _Traits> & __rhs) [inherited]`

Copies fields of `__rhs` into this.

Parameters

<code>__rhs</code>	The source values for the copies.
--------------------	-----------------------------------

Returns

Reference to this object.

All fields of `__rhs` are copied into this object except that `rdbuf()` and `rdstate()` remain unchanged. All values in the `pword` and `iword` arrays are copied. Before copying, each callback is invoked with `erase_event`. After copying, each (new) callback is invoked with `copyfmt_event`. The final step is to copy `exceptions()`.

Definition at line 63 of file `basic_ios.tcc`.

References `std::basic_ios< _CharT, _Traits>::exceptions()`, `std::basic_ios< _CharT, _Traits>::fill()`, `std::ios_base::flags()`, `std::ios_base::getloc()`, `std::ios_base::precision()`, `std::basic_ios< _CharT, _Traits>::tie()`, `std::tie()`, and `std::ios_base::width()`.

Referenced by `std::basic_ios< char, _Traits>::rdbuf()`.

5.617.5.5 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits>::eof () const [inline], [inherited]`

Fast error checking.

Returns

True if the eofbit is set.

Note that other `iostate` flags may also be set.

Definition at line 190 of file `basic_ios.h`.

5.617.5.6 `template<typename _CharT, typename _Traits> iostate std::basic_ios< _CharT, _Traits >::exceptions () const`
`[inline], [inherited]`

Throwing exceptions on errors.

Returns

The current exceptions mask.

This changes nothing in the stream. See the one-argument version of `exceptions(iostate)` for the meaning of the return value.

Definition at line 222 of file `basic_ios.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, and `std::basic_ios< char, _Traits >::setstate()`.

5.617.5.7 `template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits >::exceptions (iostate __except)`
`[inline], [inherited]`

Throwing exceptions on errors.

Parameters

<code>__except</code>	The new exceptions mask.
-----------------------	--------------------------

By default, error flags are set silently. You can set an exceptions mask for each stream; if a bit in the mask becomes set in the error flags, then an exception of type `std::ios_base::failure` is thrown.

If the error flag is already set when the exceptions mask is added, the exception is immediately thrown. Try running the following under GCC 3.1 or later:

```
1 #include <iostream>
2 #include <fstream>
3 #include <exception>
4
5 int main()
6 {
7     std::set_terminate (__gnu_cxx::__verbose_terminate_handler);
8
9     std::ifstream f ("/etc/motd");
10
11     std::cerr << "Setting badbit\n";
12     f.setstate (std::ios_base::badbit);
13
14     std::cerr << "Setting exception mask\n";
15     f.exceptions (std::ios_base::badbit);
16 }
```

Definition at line 257 of file `basic_ios.h`.

5.617.5.8 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::fail () const`
`[inline], [inherited]`

Fast error checking.

Returns

True if either the badbit or the failbit is set.

Checking the badbit in fail() is historical practice. Note that other iostate flags may also be set.

Definition at line 201 of file basic_ios.h.

Referenced by std::basic_ios< char, _Traits >::operator bool(), std::basic_ios< char, _Traits >::operator!(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_istream< _CharT, _Traits >::tellg(), std::basic_ostream< _CharT, _Traits >::tellp(), and std::regex_traits< _Ch_type >::value().

5.617.5.9 template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::fill () const
[inline], [inherited]

Retrieves the *empty* character.

Returns

The current fill character.

It defaults to a space (' ') in the current locale.

Definition at line 370 of file basic_ios.h.

Referenced by std::basic_ios< _CharT, _Traits >::copyfmt(), std::basic_ios< char, _Traits >::fill(), std::normal_distribution< _RealType >::operator()(), std::gamma_distribution< _RealType >::operator()(), std::negative_binomial_distribution< _IntType >::operator()(), and std::operator>>().

5.617.5.10 template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::fill (char_type __ch) [inline], [inherited]

Sets a new *empty* character.

Parameters

<code>__ch</code>	The new character.
-------------------	--------------------

Returns

The previous fill character.

The fill character is used to fill out space when P+ characters have been requested (e.g., via setw), Q characters are actually used, and Q<P. It defaults to a space (' ') in the current locale.

Definition at line 390 of file basic_ios.h.

5.617.5.11 `fmtflags std::ios_base::flags () const` `[inline],[inherited]`

Access to format flags.

Returns

The format control flags for both input and output.

Definition at line 619 of file `ios_base.h`.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::mersenne_twister_engine<_UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f>::discard()`, `std::money_get<_CharT, _InIter>::do_get()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_put<_CharT, _OutIter>::do_put()`, `std::subtract_with_carry_engine<_UIntType, __w, __s, __r>::operator()`, `std::discard_block_engine<_RandomNumberEngine, __p, __r>::operator()`, `std::shuffle_order_engine<_RandomNumberEngine, __k>::operator()`, `std::normal_distribution<_RealType>::operator()`, `std::gamma_distribution<_RealType>::operator()`, `std::binomial_distribution<_IntType>::operator()`, `std::negative_binomial_distribution<_IntType>::operator()`, `std::poisson_distribution<_IntType>::operator()`, `std::basic_ostream<_CharT, _Traits>::operator<<()`, `std::operator<<()`, `std::__detail::operator>>()`, `std::operator>>()`, `std::linear_congruential_engine<_UIntType, __a, __c, __m>::seed()`, and `std::basic_istream<_CharT, _Traits>::sentry::sentry()`.

5.617.5.12 `fmtflags std::ios_base::flags (fmtflags __fmtfl)` `[inline],[inherited]`

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

Definition at line 630 of file `ios_base.h`.

5.617.5.13 `template<typename _CharT, typename _Traits> streamsize std::basic_istream<_CharT, _Traits>::gcount ()`
`const` `[inline],[inherited]`

Character counting.

Returns

The number of characters extracted by the previous unformatted input function dispatched for this stream.

Definition at line 269 of file `istream`.

5.617.5.14 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits>::int_type
std::basic_istream< _CharT, _Traits>::get(void) [inherited]`

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns traits::eof().

Definition at line 236 of file istream.tcc.

References std::basic_istream< _CharT, _Traits>::_M_gcount, std::ios_base::badbit, std::ios_base::eofbit, std::ios_base::failbit, std::ios_base::goodbit, std::basic_ios< _CharT, _Traits>::rdbuf(), and std::basic_ios< _CharT, _Traits>::setstate().

Referenced by std::basic_istream< _CharT, _Traits>::get(), and std::basic_istream< _CharT, _Traits>::operator>>().

5.617.5.15 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream<
_CharT, _Traits>::get(char_type & __c) [inherited]`

Simple extraction.

Parameters

<code>__c</code>	The character in which to store data.
------------------	---------------------------------------

Returns

*this

Tries to extract a character and store it in __c. If none are available, sets failbit and returns traits::eof().

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 272 of file istream.tcc.

References std::basic_istream< _CharT, _Traits>::_M_gcount, std::ios_base::badbit, std::ios_base::eofbit, std::ios_base::failbit, std::basic_istream< _CharT, _Traits>::get(), std::ios_base::goodbit, std::basic_ios< _CharT, _Traits>::rdbuf(), and std::basic_ios< _CharT, _Traits>::setstate().

5.617.5.16 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream<
_CharT, _Traits>::get(char_type * __s, streamsize __n, char_type __delim) [inherited]`

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
<code>__n</code>	Maximum number of characters to store in <code>__s</code> .
<code>__delim</code>	A "stop" character.

Returns

*this

Characters are extracted and stored into `__s` until one of the following happens:

- `__n-1` characters are stored
- the input sequence reaches EOF
- the next character equals `__delim`, in which case the character is not extracted

If no characters are stored, failbit is set in the stream's error state.

In any case, a null character is stored into the next location in the array.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 309 of file istream.tcc.

References `std::basic_istream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::basic_istream<_CharT, _Traits>::get()`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sgetc()`, and `std::basic_streambuf<_CharT, _Traits>::snextc()`.

5.617.5.17 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::get (char_type * __s, streamsize __n) [inline], [inherited]`

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
<code>__n</code>	Maximum number of characters to store in <code>s</code> .

Returns

*this

Returns `get(__s,__n,widen("\n"))`.

Definition at line 354 of file istream.

5.617.5.18 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::get (__streambuf_type & __sb, char_type __delim) [inherited]`

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
<code>__delim</code>	A "stop" character.

Returns

*this

Characters are extracted and inserted into `__sb` until one of the following happens:

- the input sequence reaches EOF
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted)
- the next character equals `__delim` (in this case, the character is not extracted)
- an exception occurs (and in this case is caught)

If no characters are stored, failbit is set in the stream's error state.

Definition at line 356 of file istream.tcc.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sgetc()`, `std::basic_streambuf<_CharT, _Traits>::snextc()`, and `std::basic_streambuf<_CharT, _Traits>::sputc()`.

5.617.5.19 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::get (__streambuf_type & __sb) [inline], [inherited]`

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
-------------------	-------------------------------------

Returns

*this

Returns `get(__sb,widen("\n"))`.

Definition at line 387 of file `istream`.

5.617.5.20 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::getline (char_type * __s, streamsize __n, char_type __delim) [inherited]`

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.
<code>__delim</code>	A "stop" character.

Returns

*this

Extracts and stores characters into `__s` until one of the following happens. Note that these criteria are required to be tested in the order listed here, to allow an input line to exactly fill the `__s` array without setting failbit.

1. the input sequence reaches end-of-file, in which case eofbit is set in the stream error state
2. the next character equals `__delim`, in which case the character is extracted (and therefore counted in `gcount()`) but not stored
3. `__n-1` characters are stored, in which case failbit is set in the stream error state

If no characters are extracted, failbit is set. (An empty line of input should therefore not cause failbit to be set.)

In any case, a null character is stored in the next location in the array.

Definition at line 400 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_streambuf<_CharT, _Traits>::sputc()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sgetc()`, and `std::basic_streambuf<_CharT, _Traits>::snextc()`.

Referenced by `std::basic_istream<_CharT, _Traits>::get()`.

5.617.5.21 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::getline (char_type * __s, streamsize __n) [inline], [inherited]`

String extraction.

Parameters

\leftrightarrow _s	A character array in which to store the data.
\leftrightarrow _n	Maximum number of characters to extract.

Returns

*this

Returns `getline(__s,__n,widen("\n"))`.

Definition at line 427 of file istream.

5.617.5.22 `template<> basic_istream< char > & std::basic_istream< char >::getline (char_type * __s, streamsize __n, char_type __delim) [inherited]`

Explicit specialization declarations, defined in src/istream.cc.

5.617.5.23 `locale std::ios_base::getloc () const [inline],[inherited]`

Locale access.

Returns

A copy of the current locale.

If `imbue(loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale()`, the global C++ locale.

Definition at line 763 of file ios_base.h.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::money_put< _CharT, _Outlter >::do_put()`, `std::operator>>()`, and `std::ws()`.

5.617.5.24 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::good () const [inline],[inherited]`

Fast error checking.

Returns

True if no error flags are set.

A wrapper around `rdstate`.

Definition at line 180 of file basic_ios.h.

Referenced by `std::__detail::operator>>()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_istream< _CharT, _Traits >::sentry::sentry()`.

5.617.5.25 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::ignore (streamsize __n, int_type __delim) [inherited]`

Discarding characters.

Parameters

<code>__n</code>	Number of characters to discard.
<code>__delim</code>	A "stop" character.

Returns

*this

Extracts characters and throws them away until one of the following happens:

- if `__n != std::numeric_limits<int>::max()`, `__n` characters are extracted
- the input sequence reaches end-of-file
- the next character equals `__delim` (in this case, the character is extracted); note that this condition will never occur if `__delim` equals `traits::eof()`.

NB: Provide three overloads, instead of the single function (with defaults) mandated by the Standard: this leads to a better performing implementation, while still conforming to the Standard.

Definition at line 555 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_istream<_CharT, _Traits>::peek()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_streambuf<_CharT, _Traits>::sbumpc()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sgetc()`, and `std::basic_streambuf<_CharT, _Traits>::snextc()`.

5.617.5.26 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::ignore (streamsize __n) [inherited]`

Simple extraction.

Returns

A character, or `eof()`.

Tries to extract a character. If none are available, sets failbit and returns `traits::eof()`.

Definition at line 493 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sgetc()`, and `std::basic_streambuf<_CharT, _Traits>::snextc()`.

5.617.5.27 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::ignore (void) [inherited]`

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns traits::eof().

Definition at line 460 of file istream.tcc.

References std::basic_istream< _CharT, _Traits>::_M_gcount, std::ios_base::badbit, std::ios_base::eofbit, std::ios_base::goodbit, std::basic_ios< _CharT, _Traits>::rdbuf(), std::basic_streambuf< _CharT, _Traits>::sbumpc(), and std::basic_ios< _CharT, _Traits>::setstate().

Referenced by std::basic_istream< _CharT, _Traits>::getline(), and std::basic_istream< _CharT, _Traits>::ignore().

5.617.5.28 `template<typename _CharT, typename _Traits> locale std::basic_ios< _CharT, _Traits>::imbue (const locale & __loc) [inherited]`

Moves to a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls `ios_base::imbue(loc)`, and if a stream buffer is associated with this stream, calls that buffer's `pubimbue(loc)`.

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Definition at line 114 of file basic_ios.tcc.

References std::ios_base::imbue().

Referenced by std::basic_ios< char, _Traits>::fill().

5.617.5.29 `template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits>::init (basic_streambuf< _CharT, _Traits> * __sb) [protected], [inherited]`

All setup is performed here.

This is called from the public constructor. It is not virtual and cannot be redefined.

Definition at line 126 of file basic_ios.tcc.

Referenced by std::basic_ios< char, _Traits>::basic_ios().

5.617.5.30 `long& std::ios_base::iword (int __ix) [inline],[inherited]`

Access to integer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to an integer associated with the index.

The `iword` function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 809 of file `ios_base.h`.

5.617.5.31 `template<typename _CharT, typename _Traits> char std::basic_ios<_CharT, _Traits>::narrow (char_type __c, char __default) const [inline],[inherited]`

Squeezes characters.

Parameters

<code>__c</code>	The character to narrow.
<code>__default</code>	The character to narrow.

Returns

The narrowed character.

Maps a character of `char_type` to a character of `char`, if possible.

Returns the result of

```
1 std::use_facet<ctype<char_type>> >(getloc()).narrow(c,default)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Definition at line 430 of file `basic_ios.h`.

```
5.617.5.32 template<typename _CharT, typename _Traits> std::basic_ios<_CharT, _Traits>::operator bool ( ) const
[inline], [explicit], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`.

Definition at line 117 of file `basic_ios.h`.

```
5.617.5.33 template<typename _CharT, typename _Traits> bool std::basic_ios<_CharT, _Traits>::operator! ( ) const
[inline], [inherited]
```

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`.

Definition at line 125 of file `basic_ios.h`.

```
5.617.5.34 template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>
::operator>> ( __istream_type &(*)(__istream_type &)__pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

Definition at line 120 of file `istream`.

Referenced by `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::basic_istream<_CharT, _Traits>::sentry←::sentry()`, and `std::ws()`.

```
5.617.5.35 template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>
::operator>> ( __ios_type &(*)(__ios_type &)__pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

Definition at line 124 of file `istream`.

```
5.617.5.36 template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>
::operator>> ( ios_base &(*)(ios_base &)__pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

Definition at line 131 of file `istream`.

```
5.617.5.37 template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>
::operator>> ( bool &__n ) [inline], [inherited]
```

Integer arithmetic extractors.

Parameters

<code>_↔</code> <code>_n</code>	A variable of builtin integral type.
------------------------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 168 of file `istream`.

5.617.5.38 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::operator>> (short & __n) [inherited]`

Integer arithmetic extractors.

Parameters

<code>_↔</code> <code>_n</code>	A variable of builtin integral type.
------------------------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 114 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::failbit`, `std::num_get< _CharT, _InIter>::get()`, `std::ios_base::goodbit`, `std::basic_istream< _CharT, _Traits>::operator>>()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

5.617.5.39 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::operator>> (unsigned short & __n) [inline], [inherited]`

Integer arithmetic extractors.

Parameters

<code>_↔</code> <code>_n</code>	A variable of builtin integral type.
------------------------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 175 of file `istream`.

5.617.5.40 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::operator>> (int &__n) [inherited]`

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 159 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::failbit`, `std::num_get<_CharT, _InIter>::get()`, `std::ios_base::goodbit`, `std::basic_istream<_CharT, _Traits>::operator>>()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

5.617.5.41 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (unsigned int &__n) [inline], [inherited]`

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 182 of file `istream`.

5.617.5.42 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::operator>> (long & __n) [inline], [inherited]`

Integer arithmetic extractors.

Parameters

<code>_↔</code>	A variable of builtin integral type.
<code>_n</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 186 of file `istream`.

```
5.617.5.43 template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> ( unsigned long &_n ) [inline],[inherited]
```

Integer arithmetic extractors.

Parameters

<code>_↔</code>	A variable of builtin integral type.
<code>_n</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 190 of file `istream`.

```
5.617.5.44 template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> ( long long &_n ) [inline],[inherited]
```

Integer arithmetic extractors.

Parameters

<code>_↔</code>	A variable of builtin integral type.
<code>_n</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 195 of file `istream`.

5.617.5.45 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::operator>> (unsigned long long & __n) [inline], [inherited]`

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 199 of file `istream`.

5.617.5.46 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::operator>> (float & __f) [inline], [inherited]`

Floating point arithmetic extractors.

Parameters

<code>__f</code>	A variable of builtin floating point type.
------------------	--

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 214 of file `istream`.

5.617.5.47 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::operator>> (double & __f) [inline], [inherited]`

Floating point arithmetic extractors.

Parameters

↔	A variable of builtin floating point type.
↔	
↔	
↔	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 218 of file `istream`.

5.617.5.48 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (long double &__f) [inline], [inherited]`

Floating point arithmetic extractors.

Parameters

↔	A variable of builtin floating point type.
↔	
↔	
↔	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 222 of file `istream`.

5.617.5.49 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (void *&__p) [inline], [inherited]`

Basic arithmetic extractors.

Parameters

↔	A variable of pointer type.
<i>p</i>	

Returns

*`this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 235 of file `istream`.

5.617.5.50 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::operator>> (__streambuf_type* __sb) [inherited]`

Extracting into another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is `NULL`, the stream will set `failbit` in its error state.

Characters are extracted from this stream and inserted into the `__sb` streambuf until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs (and in this case is caught)

If the function inserts no characters, `failbit` is set.

Definition at line 204 of file `istream.tcc`.

References `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::basic_istream<_CharT, _Traits>::get()`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

5.617.5.51 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits>::int_type std::basic_istream<_CharT, _Traits>::peek (void) [inherited]`

Looking ahead in the stream.

Returns

The next character, or `eof()`.

If, after constructing the sentry object, `good()` is `false`, returns `traits::eof()`. Otherwise reads but does not extract the next input character.

Definition at line 620 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_istream<_CharT, _Traits>::read()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream<_CharT, _Traits>::ignore()`.

5.617.5.52 **streamsize** std::ios_base::precision () const [inline],[inherited]

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Definition at line 689 of file ios_base.h.

Referenced by std::basic_ios<_CharT, _Traits>::copyfmt(), std::num_get<_CharT, _InIter>::do_get(), std::normal_↵distribution<_RealType>::operator>(), std::gamma_distribution<_RealType>::operator>(), std::negative_binomial_↵distribution<_IntType>::operator>(), and std::operator>>().

5.617.5.53 **streamsize** std::ios_base::precision (streamsize __prec) [inline],[inherited]

Changing flags.

Parameters

<code>__prec</code>	The new precision value.
---------------------	--------------------------

Returns

The previous value of precision().

Definition at line 698 of file ios_base.h.

5.617.5.54 **template**<typename _CharT, typename _Traits> **basic_istream**<_CharT, _Traits> &std::basic_istream<_CharT, _Traits>::putback (char_type __c) [inherited]

Unextracting a single character.

Parameters

<code>__c</code>	The character to push back into the input stream.
------------------	---

Returns

*this

If rdbuf () is not null, calls rdbuf ()->sputbackc (c) .

If rdbuf () is null or if sputbackc () fails, sets badbit in the error state.

Note

This function first clears eofbit. Since no characters are extracted, the next call to `gcount()` will return 0, as required by DR 60.

Definition at line 711 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::rdstate()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sputbackc()`, and `std::basic_istream<_CharT, _Traits>::unget()`.

Referenced by `std::basic_istream<_CharT, _Traits>::readsome()`.

5.617.5.55 `void*& std::ios_base::pword (int __ix)` `[inline]`, `[inherited]`

Access to void pointer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to a `void*` associated with the index.

The `pword` function provides access to an array of pointers that can be used for any purpose. The array grows as required to hold the supplied index. All pointers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 830 of file `ios_base.h`.

5.617.5.56 `template<typename _CharT, typename _Traits> basic_streambuf<_CharT, _Traits> * std::basic_ios<_CharT, _Traits>::rdbuf (basic_streambuf<_CharT, _Traits> * __sb)` `[inherited]`

Changing the underlying buffer.

Parameters

<code>__sb</code>	The new stream buffer.
-------------------	------------------------

Returns

The previous stream buffer.

Associates a new buffer with the current stream, and clears the error state.

Due to historical accidents which the LWG refuses to correct, the I/O library suffers from a design error: this function is hidden in derived classes by overrides of the zero-argument `rdbuf()`, which is non-virtual for hysterical raisins. As a result, you must use explicit qualifications to access this function via any derived class. For example:

```
1 std::fstream    foo;           // or some other derived type
2 std::streambuf* p = .....;
3
4 foo.ios::rdbuf(p);           // ios == basic_ios<char>
```

Definition at line 53 of file `basic_ios.tcc`.

5.617.5.57 `template<typename _CharT, typename _Traits, typename _Alloc> __stringbuf_type* std::basic_istream<_CharT, _Traits, _Alloc>::rdbuf() const [inline]`

Accessing the underlying buffer.

Returns

The current `basic_stringbuf` buffer.

This hides both signatures of `std::basic_ios::rdbuf()`.

Definition at line 472 of file `sstream`.

5.617.5.58 `template<typename _CharT, typename _Traits> iosstate std::basic_ios<_CharT, _Traits>::rdstate() const [inline], [inherited]`

Returns the error state of the stream buffer.

Returns

A bit pattern (well, isn't everything?)

See `std::ios_base::iosstate` for the possible bit values. Most users will call one of the interpreting wrappers, e.g., `good()`.

Definition at line 137 of file `basic_ios.h`.

Referenced by `std::basic_ios<char, _Traits>::bad()`, `std::basic_ios<char, _Traits>::eof()`, `std::basic_ios<char, _Traits>::fail()`, `std::basic_ios<char, _Traits>::good()`, `std::basic_istream<_CharT, _Traits>::putback()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_ios<char, _Traits>::setstate()`, and `std::basic_istream<_CharT, _Traits>::unget()`.

5.617.5.59 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::read(char_type* __s, streamsize __n) [inherited]`

Extraction without delimiters.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

`*this`

If the stream state is `good()`, extracts characters and stores them into `__s` until one of the following happens:

- `__n` characters are stored
- the input sequence reaches end-of-file, in which case the error state is set to `failbit|eofbit`.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 650 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_istream<_CharT, _Traits>::readsome()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream<_CharT, _Traits>::peek()`.

5.617.5.60 `template<typename _CharT, typename _Traits> streamsize std::basic_istream<_CharT, _Traits>::readsome (char_type* __s, streamsize __n) [inherited]`

Extraction until the buffer is exhausted, but no more.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

The number of characters extracted.

Extracts characters and stores them into `__s` depending on the number of characters remaining in the streambuf's buffer, `rdbuf()->in_avail()`, called A here:

- if $A == -1$, sets eofbit and extracts no characters
- if $A == 0$, extracts no characters
- if $A > 0$, extracts $\min(A, n)$

The goal is to empty the current buffer, and to not request any more from the external input sequence controlled by the streambuf.

Definition at line 679 of file istream.tcc.

References std::basic_istream< _CharT, _Traits >::M_gcount, std::ios_base::badbit, std::ios_base::eofbit, std::ios_base::goodbit, std::min(), std::basic_istream< _CharT, _Traits >::putback(), std::basic_ios< _CharT, _Traits >::rdbuf(), and std::basic_ios< _CharT, _Traits >::setstate().

Referenced by std::basic_istream< _CharT, _Traits >::read().

5.617.5.61 void std::ios_base::register_callback (event_callback __fn, int __index) [inherited]

Add the callback __fn with parameter __index.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

5.617.5.62 template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::seekg (pos_type __pos) [inherited]

Changing the current read position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

*this

If fail() is not true, calls rdbuf()->pubseekpos(__pos). If that function fails, sets failbit.

Note

This function first clears eofbit. It does not count the number of characters extracted, if any, and therefore does not affect the next call to gcount().

Definition at line 845 of file istream.tcc.

References `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::ios_base::in`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::rdstate()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream<_CharT, _Traits>::tellg()`.

5.617.5.63 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::seekg (off_type __off, ios_base::seekdir __dir) [inherited]`

Changing the current read position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

`*this`

If `fail()` is not true, calls `rdbuf()->pubseekoff(__off, __dir)`. If that function fails, sets `failbit`.

Note

This function first clears `eofbit`. It does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

Definition at line 884 of file istream.tcc.

References `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::ios_base::in`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::rdstate()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

5.617.5.64 `fmtflags std::ios_base::setf (fmtflags __fmtfl) [inline], [inherited]`

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
----------------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set.

Definition at line 646 of file ios_base.h.

Referenced by std::boolalpha(), std::dec(), std::fixed(), std::hex(), std::hexfloat(), std::left(), std::oct(), std::__detail::operator>>(), std::right(), std::scientific(), std::showbase(), std::showpoint(), std::showpos(), std::skipws(), std::unitbuf(), and std::uppercase().

5.617.5.65 `fmtflags std::ios_base::setf (fmtflags __fmtfl, fmtflags __mask)` `[inline], [inherited]`

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
<code>__mask</code>	The flags mask for <i>fmtfl</i> .

Returns

The previous format control flags.

This function clears *mask* in the format flags, then sets *fmtfl* & *mask*. An example mask is `ios_base::adjustfield`.

Definition at line 663 of file ios_base.h.

5.617.5.66 `template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits >::setstate (iostate __state)` `[inline], [inherited]`

Sets additional flags in the error state.

Parameters

<code>__state</code>	The additional state flag(s) to set.
----------------------	--------------------------------------

See `std::ios_base::iostate` for the possible bit values.

Definition at line 157 of file basic_ios.h.

Referenced by `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::tr2::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::ws()`.

5.617.5.67 `template<typename _CharT, typename _Traits, typename _Alloc> __string_type std::basic_istream<_CharT, _Traits, _Alloc>::str () const [inline]`

Copying out the string buffer.

Returns

`rdbuf ()->str ()`

Definition at line 480 of file sstream.

5.617.5.68 `template<typename _CharT, typename _Traits, typename _Alloc> void std::basic_istream<_CharT, _Traits, _Alloc>::str (const __string_type & __s) [inline]`

Setting a new buffer.

Parameters

<code>__s</code>	The string to use as a new sequence.
------------------	--------------------------------------

Calls `rdbuf ()->str (s)`.

Definition at line 490 of file sstream.

5.617.5.69 `template<typename _CharT, typename _Traits> int std::basic_istream<_CharT, _Traits>::sync (void) [inherited]`

Synchronizing the stream buffer.

Returns

0 on success, -1 on failure

If `rdbuf ()` is a null pointer, returns -1.

Otherwise, calls `rdbuf ()->pubsync ()`, and if that returns -1, sets badbit and returns -1.

Otherwise, returns 0.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount ()`.

Definition at line 781 of file istream.tcc.

References `std::ios_base::badbit`, `std::ios_base::goodbit`, `std::basic_streambuf<_CharT, _Traits>::pubsync()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::setstate()`, and `std::basic_istream<_CharT, _Traits>::tellg()`.

Referenced by `std::basic_istream<_CharT, _Traits>::unget()`.

5.617.5.70 `static bool std::ios_base::sync_with_stdio (bool __sync = true) [static], [inherited]`

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., stdout) and the standard C++ objects (e.g., cout). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstream.html#std.io.filestreams.binary>

5.617.5.71 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits>::pos_type
std::basic_istream<_CharT, _Traits>::tellg(void) [inherited]`

Getting the current read position.

Returns

A file position object.

If `fail()` is not false, returns `pos_type(-1)` to indicate failure. Otherwise returns `rdbuf()->pubseekoff(0, cur, in)`.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`. At variance with `putback`, `unget` and `seekg`, `eofbit` is not cleared first.

Definition at line 817 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::cur`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::ios_base::in`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_istream<_CharT, _Traits>::seekg()`.

Referenced by `std::basic_istream<_CharT, _Traits>::sync()`.

5.617.5.72 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>* std::basic_ios<_CharT,
_Traits>::tie() const [inline], [inherited]`

Fetches the current *tied* stream.

Returns

A pointer to the tied stream, or NULL if the stream is not tied.

A stream may be *tied* (or synchronized) to a second output stream. When this stream performs any I/O, the tied stream is first flushed. For example, `std::cin` is tied to `std::cout`.

Definition at line 295 of file `basic_ios.h`.

Referenced by `std::basic_ios<char, _Traits>::basic_ios()`, `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`, and `std::basic_istream<_CharT, _Traits>::sentry::sentry()`.

5.617.5.73 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>* std::basic_ios<_CharT,
_Traits>::tie(basic_ostream<_CharT, _Traits>* __tiestr) [inline], [inherited]`

Ties this stream to an output stream.

Parameters

<code>__tiestr</code>	The output stream.
-----------------------	--------------------

Returns

The previously tied output stream, or NULL if the stream was not tied.

This sets up a new tie; see `tie()` for more.

Definition at line 307 of file `basic_ios.h`.

5.617.5.74 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::unget(void) [inherited]`

Unextracting the previous character.

Returns

`*this`

If `rdbuf()` is not null, calls `rdbuf()->sungetc()`.

If `rdbuf()` is null or if `sungetc()` fails, sets `badbit` in the error state.

Note

This function first clears `eofbit`. Since no characters are extracted, the next call to `gcount()` will return 0, as required by DR 60.

Definition at line 746 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::rdstate()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sungetc()`, and `std::basic_istream<_CharT, _Traits>::sync()`.

Referenced by `std::__detail::operator>>()`, and `std::basic_istream<_CharT, _Traits>::putback()`.

5.617.5.75 `void std::ios_base::unsetf(fmtflags __mask) [inline], [inherited]`

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

Definition at line 678 of file `ios_base.h`.

Referenced by `std::defaultfloat()`, `std::noboolalpha()`, `std::noshowbase()`, `std::noshowpoint()`, `std::noshowpos()`, `std::noskipws()`, `std::nounitbuf()`, and `std::nouppercase()`.

5.617.5.76 `template<typename _CharT, typename _Traits> char_type std::basic_ios<_CharT, _Traits>::widen (char __c)`
`const [inline],[inherited]`

Widens characters.

Parameters

<code>__c</code>	The character to widen.
------------------	-------------------------

Returns

The widened character.

Maps a character of `char` to a character of `char_type`.

Returns the result of

```
1 std::use_facet<ctype<char_type>> >(getloc()).widen(c)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Definition at line 449 of file `basic_ios.h`.

Referenced by `std::basic_ios<char, _Traits>::fill()`, `std::getline()`, `std::normal_distribution<_RealType>::operator()()`, `std::gamma_distribution<_RealType>::operator()()`, `std::negative_binomial_distribution<_IntType>::operator()()`, `std::tr2::operator>>()`, and `std::operator>>()`.

5.617.5.77 `streamsize std::ios_base::width () const [inline],[inherited]`

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Definition at line 712 of file `ios_base.h`.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::money_get<_CharT, _InIter>::do_get()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_put<_CharT, _OutIter>::do_put()`, and `std::operator>>()`.

5.617.5.78 `streamsize std::ios_base::width (streamsize __wide) [inline],[inherited]`

Changing flags.

Parameters

<code>__wide</code>	The new width value.
---------------------	----------------------

Returns

The previous value of `width()`.

Definition at line 721 of file `ios_base.h`.

5.617.5.79 `static int std::ios_base::xalloc () throw` `[static], [inherited]`

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the `iwor` and `pwor` functions. The expectation is that an application calls `xalloc` in order to obtain an index in the `iwor` and `pwor` arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. `xalloc` is guaranteed to return an index that is safe to use in the `iwor` and `pwor` arrays.

5.617.6 Member Data Documentation

5.617.6.1 `template<typename _CharT, typename _Traits> streamsize std::basic_istream< _CharT, _Traits >::_M_gcount`
`[protected], [inherited]`

The number of characters extracted in the previous unformatted function; see `gcount()`.

Definition at line 82 of file `istream`.

Referenced by `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

5.617.6.2 `const fmtflags std::ios_base::adjustfield` `[static], [inherited]`

A mask of left|right|internal. Useful for the 2-arg form of `setf`.

Definition at line 378 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::internal()`, `std::left()`, and `std::right()`.

5.617.6.3 `const openmode std::ios_base::app` `[static]`, `[inherited]`

Seek to end before each write.

Definition at line 432 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

5.617.6.4 `const openmode std::ios_base::ate` `[static]`, `[inherited]`

Open and seek to end immediately after opening.

Definition at line 435 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::open()`.

5.617.6.5 `const iostate std::ios_base::badbit` `[static]`, `[inherited]`

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Definition at line 402 of file `ios_base.h`.

Referenced by `std::basic_ios< char, _Traits >::bad()`, `std::basic_ios< char, _Traits >::fail()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsomewhat()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::basic_ostream< _CharT, _Traits >::write()`.

5.617.6.6 `const fmtflags std::ios_base::basefield` `[static]`, `[inherited]`

A mask of `dec|oct|hex`. Useful for the 2-arg form of `setf`.

Definition at line 381 of file `ios_base.h`.

Referenced by `std::dec()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::hex()`, `std::oct()`, and `std::basic_ostream< _CharT, _Traits >::operator<<()`.

5.617.6.7 `const seekdir std::ios_base::beg` `[static]`, `[inherited]`

Request a seek relative to the beginning of the stream.

Definition at line 464 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::seekpos()`, and `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync()`.

5.617.6.8 const openmode std::ios_base::binary [static],[inherited]

Perform input and output in binary mode (as opposed to text mode). This is probably not what you think it is; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.binary>.

Definition at line 440 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::showmanyc().

5.617.6.9 const fmtflags std::ios_base::boolalpha [static],[inherited]

Insert/extract `bool` in alphabetic rather than numeric format.

Definition at line 326 of file ios_base.h.

Referenced by std::boolalpha(), std::num_get< _CharT, _InIter >::do_get(), std::num_put< _CharT, _OutIter >::do_put(), and std::noboolalpha().

5.617.6.10 const seekdir std::ios_base::cur [static],[inherited]

Request a seek relative to the current position within the sequence.

Definition at line 467 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::imbue(), std::basic_filebuf< _CharT, _Traits >::overflow(), std::basic_filebuf< _CharT, _Traits >::pbackfail(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff(), std::basic_filebuf< _CharT, _Traits >::seekoff(), __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync(), std::basic_istream< _CharT, _Traits >::tellg(), and std::basic_ostream< _CharT, _Traits >::tellp().

5.617.6.11 const fmtflags std::ios_base::dec [static],[inherited]

Converts integer input or generates integer output in decimal base.

Definition at line 329 of file ios_base.h.

Referenced by std::dec().

5.617.6.12 const seekdir std::ios_base::end [static],[inherited]

Request a seek relative to the current end of the sequence.

Definition at line 470 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::open(), and std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff().

5.617.6.13 const iostate std::ios_base::eofbit [static],[inherited]

Indicates that an input operation reached the end of an input sequence.

Definition at line 405 of file ios_base.h.

Referenced by std::time_get< _CharT, _InIter >::do_get(), std::num_get< _CharT, _InIter >::do_get(), std::time_get< _CharT, _InIter >::do_get_date(), std::time_get< _CharT, _InIter >::do_get_monthname(), std::time_get< _CharT, _InIter >::do_get_time(), std::time_get< _CharT, _InIter >::do_get_weekday(), std::time_get< _CharT, _InIter >::do_get_year(), std::basic_ios< char, _Traits >::eof(), std::basic_istream< _CharT, _Traits >::get(), std::time_get< _CharT, _InIter >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::basic_istream< _CharT, _Traits >::ignore(), std::basic_istream< _CharT, _Traits >::operator>>(), std::operator>>(), std::basic_istream< _CharT, _Traits >::peek(), std::basic_istream< _CharT, _Traits >::putback(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::readsome(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_istream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::unget(), and std::ws().

5.617.6.14 const iostate std::ios_base::failbit [static],[inherited]

Indicates that an input operation failed to read the expected characters, or that an output operation failed to generate the desired characters.

Definition at line 410 of file ios_base.h.

Referenced by std::time_get< _CharT, _InIter >::do_date_order(), std::num_get< _CharT, _InIter >::do_get(), std::time_get< _CharT, _InIter >::do_get_monthname(), std::time_get< _CharT, _InIter >::do_get_weekday(), std::time_get< _CharT, _InIter >::do_get_year(), std::basic_ios< char, _Traits >::fail(), std::basic_istream< _CharT, _Traits >::get(), std::time_get< _CharT, _InIter >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::basic_ostream< _CharT, _Traits >::operator<<(), std::basic_istream< _CharT, _Traits >::operator>>(), std::operator>>(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_ostream< _CharT, _Traits >::sentry::sentry(), and std::basic_istream< _CharT, _Traits >::sentry::sentry().

5.617.6.15 const fmtflags std::ios_base::fixed [static],[inherited]

Generate floating-point output in fixed-point notation.

Definition at line 332 of file ios_base.h.

Referenced by std::num_get< _CharT, _InIter >::do_get(), std::fixed(), and std::hexfloat().

5.617.6.16 const fmtflags std::ios_base::floatfield [static],[inherited]

A mask of scientific|fixed. Useful for the 2-arg form of setf.

Definition at line 384 of file ios_base.h.

Referenced by std::defaultfloat(), std::num_get< _CharT, _InIter >::do_get(), std::fixed(), std::hexfloat(), and std::scientific().

5.617.6.17 `const iostate std::ios_base::goodbit` `[static], [inherited]`

Indicates all is well.

Definition at line 413 of file `ios_base.h`.

Referenced by `std::time_get< _CharT, _InIter >::do_date_order()`, `std::time_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::time_get< _CharT, _InIter >::do_get_monthname()`, `std::time_get< _CharT, _InIter >::do_get_weekday()`, `std::time_get< _CharT, _InIter >::do_get_year()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::time_get< _CharT, _InIter >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_ios< char, _Traits >::rdstate()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sync()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

5.617.6.18 `const fmtflags std::ios_base::hex` `[static], [inherited]`

Converts integer input or generates integer output in hexadecimal base.

Definition at line 335 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::hex()`, and `std::basic_ostream< _CharT, _Traits >::operator<<()`.

5.617.6.19 `const openmode std::ios_base::in` `[static], [inherited]`

Open for input. Default for `ifstream` and `fstream`.

Definition at line 443 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos()`, `std::basic_filebuf< _CharT, _Traits >::showmanyc()`, `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::underflow()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

5.617.6.20 `const fmtflags std::ios_base::internal` `[static], [inherited]`

Adds fill characters at a designated internal point in certain generated output, or identical to `right` if no such point is designated.

Definition at line 340 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, and `std::internal()`.

5.617.6.21 const fmtflags std::ios_base::left [static], [inherited]

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.)

Definition at line 344 of file ios_base.h.

Referenced by std::money_get< _CharT, _InIter >::do_get(), std::num_put< _CharT, _OutIter >::do_put(), and std::left().

5.617.6.22 const fmtflags std::ios_base::oct [static], [inherited]

Converts integer input or generates integer output in octal base.

Definition at line 347 of file ios_base.h.

Referenced by std::num_get< _CharT, _InIter >::do_get(), std::oct(), and std::basic_ostream< _CharT, _Traits >::operator<<().

5.617.6.23 const openmode std::ios_base::out [static], [inherited]

Open for output. Default for ofstream and fstream.

Definition at line 446 of file ios_base.h.

Referenced by std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow(), std::basic_filebuf< _CharT, _Traits >::overflow(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::pbackfail(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos(), __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync(), std::basic_ostream< _CharT, _Traits >::tellp(), and std::basic_filebuf< _CharT, _Traits >::xsputn().

5.617.6.24 const fmtflags std::ios_base::right [static], [inherited]

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Definition at line 351 of file ios_base.h.

Referenced by std::right().

5.617.6.25 const fmtflags std::ios_base::scientific [static], [inherited]

Generates floating-point output in scientific notation.

Definition at line 354 of file ios_base.h.

Referenced by std::hexfloat(), and std::scientific().

5.617.6.26 const fmtflags std::ios_base::showbase [static], [inherited]

Generates a prefix indicating the numeric base of generated integer output.

Definition at line 358 of file ios_base.h.

Referenced by std::money_get< _CharT, _InIter >::do_get(), std::num_get< _CharT, _InIter >::do_get(), std::num_put< _CharT, _OutIter >::do_put(), std::noshowbase(), and std::showbase().

5.617.6.27 `const fmtflags std::ios_base::showpoint` `[static], [inherited]`

Generates a decimal-point character unconditionally in generated floating-point output.

Definition at line 362 of file `ios_base.h`.

Referenced by `std::noshowpoint()`, and `std::showpoint()`.

5.617.6.28 `const fmtflags std::ios_base::showpos` `[static], [inherited]`

Generates a + sign in non-negative generated numeric output.

Definition at line 365 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _Inlter >::do_get()`, `std::noshowpos()`, and `std::showpos()`.

5.617.6.29 `const fmtflags std::ios_base::skipws` `[static], [inherited]`

Skips leading white space before certain input operations.

Definition at line 368 of file `ios_base.h`.

Referenced by `std::noskipws()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, and `std::skipws()`.

5.617.6.30 `const openmode std::ios_base::trunc` `[static], [inherited]`

Open for input. Default for `ofstream`.

Definition at line 449 of file `ios_base.h`.

5.617.6.31 `const fmtflags std::ios_base::unitbuf` `[static], [inherited]`

Flushes output after each output operation.

Definition at line 371 of file `ios_base.h`.

Referenced by `std::nunitbuf()`, and `std::unitbuf()`.

5.617.6.32 `const fmtflags std::ios_base::uppercase` `[static], [inherited]`

Replaces certain lowercase letters with their uppercase equivalents in generated output.

Definition at line 375 of file `ios_base.h`.

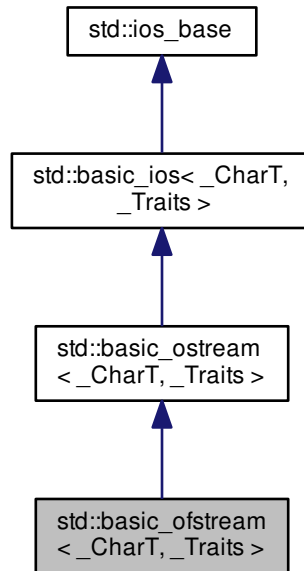
Referenced by `std::num_get< _CharT, _Inlter >::do_get()`, `std::num_put< _CharT, _Outlter >::do_put()`, `std::noskipws()`, and `std::uppercase()`.

The documentation for this class was generated from the following files:

- [iosfwd](#)
- [sstream](#)

5.618 std::basic_ofstream< _CharT, _Traits > Class Template Reference

Inheritance diagram for std::basic_ofstream< _CharT, _Traits >:



Public Types

- typedef [ctype](#)< _CharT > **__ctype_type**
- typedef [basic_filebuf](#)< char_type, traits_type > **__filebuf_type**
- typedef [basic_ios](#)< _CharT, _Traits > **__ios_type**
- typedef [num_put](#)< _CharT, [ostreambuf_iterator](#)< _CharT, _Traits > > **__num_put_type**
- typedef [basic_ostream](#)< char_type, traits_type > **__ostream_type**
- typedef [basic_streambuf](#)< _CharT, _Traits > **__streambuf_type**
- typedef _CharT **char_type**
- enum [event](#) { [erase_event](#), [imbue_event](#), [copyfmt_event](#) }
- typedef void(* [event_callback](#)) (event __e, [ios_base](#) & __b, int __i)
- typedef _ios_Fmtflags [fmtflags](#)
- typedef traits_type::int_type **int_type**
- typedef int **io_state**
- typedef _ios_istate [iostate](#)
- typedef traits_type::off_type **off_type**
- typedef int **open_mode**
- typedef _ios_Openmode [openmode](#)
- typedef traits_type::pos_type **pos_type**
- typedef int **seek_dir**
- typedef _ios_Seekdir [seekdir](#)

- typedef [std::streamoff](#) **streamoff**
- typedef [std::streampos](#) **streampos**
- typedef [_Traits](#) **traits_type**
- typedef [num_get<_CharT, istreambuf_iterator<_CharT, _Traits>>](#) **__num_get_type**

Public Member Functions

- [basic_ofstream](#) ()
- [basic_ofstream](#) (const char *__s, [ios_base::openmode](#) __mode=[ios_base::out|ios_base::trunc](#))
- [basic_ofstream](#) (const [std::string](#) &__s, [ios_base::openmode](#) __mode=[ios_base::out|ios_base::trunc](#))
- **basic_ofstream** (const [basic_ofstream](#) &)=delete
- **basic_ofstream** ([basic_ofstream](#) &&__rhs)
- [~basic_ofstream](#) ()
- const [locale](#) & [_M_getloc](#) () const
- template<typename [_ValueT](#) >
 [basic_ostream](#)< [_CharT](#), [_Traits](#) > & [_M_insert](#) ([_ValueT](#) __v)
- void [_M_setstate](#) ([iostate](#) __state)
- bool [bad](#) () const
- void [clear](#) ([iostate](#) __state=[goodbit](#))
- void [close](#) ()
- [basic_ios](#) & [copyfmt](#) (const [basic_ios](#) &__rhs)
- bool [eof](#) () const
- [iostate](#) [exceptions](#) () const
- void [exceptions](#) ([iostate](#) __except)
- bool [fail](#) () const
- char_type [fill](#) () const
- char_type [fill](#) (char_type __ch)
- [fmtflags](#) [flags](#) () const
- [fmtflags](#) [flags](#) ([fmtflags](#) __fmtfl)
- [__ostream_type](#) & [flush](#) ()
- [locale](#) [getloc](#) () const
- bool [good](#) () const
- [locale](#) [imbue](#) (const [locale](#) &__loc)
- bool [is_open](#) ()
- bool [is_open](#) () const
- long & [iword](#) (int __ix)
- char [narrow](#) (char_type __c, char __dfault) const
- void [open](#) (const char *__s, [ios_base::openmode](#) __mode=[ios_base::out|ios_base::trunc](#))
- void [open](#) (const [std::string](#) &__s, [ios_base::openmode](#) __mode=[ios_base::out|ios_base::trunc](#))
- [__ostream_type](#) & [operator<<](#) (const void *__p)
- [__ostream_type](#) & [operator<<](#) ([__streambuf_type](#) *__sb)
- [basic_ofstream](#) & **operator=** (const [basic_ofstream](#) &)=delete
- [basic_ofstream](#) & **operator=** ([basic_ofstream](#) &&__rhs)
- [streamsize](#) [precision](#) () const
- [streamsize](#) [precision](#) ([streamsize](#) __prec)
- void *& [pword](#) (int __ix)
- [basic_streambuf](#)< [_CharT](#), [_Traits](#) > * [rdbuf](#) ([basic_streambuf](#)< [_CharT](#), [_Traits](#) > *__sb)
- [__filebuf_type](#) * [rdbuf](#) () const

- `iosstate rdstate () const`
- `void register_callback (event_callback __fn, int __index)`
- `__ostream_type & seekp (pos_type)`
- `__ostream_type & seekp (off_type, ios_base::seekdir)`
- `fmtflags setf (fmtflags __fmtfl)`
- `fmtflags setf (fmtflags __fmtfl, fmtflags __mask)`
- `void setstate (iosstate __state)`
- `void swap (basic_ofstream &__rhs)`
- `pos_type tellp ()`
- `basic_ofstream< _CharT, _Traits > * tie () const`
- `basic_ofstream< _CharT, _Traits > * tie (basic_ofstream< _CharT, _Traits > *__tiestr)`
- `void unsetf (fmtflags __mask)`
- `char_type widen (char __c) const`
- `streamsize width () const`
- `streamsize width (streamsize __wide)`
- `__ostream_type & operator<< (__ostream_type &(__pf)(__ostream_type &))`
- `__ostream_type & operator<< (__ios_type &(__pf)(__ios_type &))`
- `__ostream_type & operator<< (ios_base &(__pf)(ios_base &))`

Inserters

All the `operator<<` functions (aka formatted output functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_ofstream::sentry`. This can have several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- `__ostream_type & operator<< (long __n)`
- `__ostream_type & operator<< (unsigned long __n)`
- `__ostream_type & operator<< (bool __n)`
- `__ostream_type & operator<< (short __n)`
- `__ostream_type & operator<< (unsigned short __n)`
- `__ostream_type & operator<< (int __n)`
- `__ostream_type & operator<< (unsigned int __n)`
- `__ostream_type & operator<< (long long __n)`
- `__ostream_type & operator<< (unsigned long long __n)`
- `__ostream_type & operator<< (double __f)`
- `__ostream_type & operator<< (float __f)`
- `__ostream_type & operator<< (long double __f)`

Unformatted Output Functions

All the unformatted output functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_ofstream::sentry`. This has several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state. If `badbit` is on in the stream's exceptions mask, the exception will be rethrown without completing its actions.

- `__ostream_type & put (char_type __c)`
- `void M_write (const char_type *__s, streamsize __n)`
- `__ostream_type & write (const char_type *__s, streamsize __n)`
- `operator bool () const`
- `bool operator! () const`

Static Public Member Functions

- static bool [sync_with_stdio](#) (bool __sync=true)
- static int [xalloc](#) () throw ()

Static Public Attributes

- static const [fmtflags](#) [adjustfield](#)
- static const [openmode](#) [app](#)
- static const [openmode](#) [ate](#)
- static const [iostate](#) [badbit](#)
- static const [fmtflags](#) [basefield](#)
- static const [seekdir](#) [beg](#)
- static const [openmode](#) [binary](#)
- static const [fmtflags](#) [boolalpha](#)
- static const [seekdir](#) [cur](#)
- static const [fmtflags](#) [dec](#)
- static const [seekdir](#) [end](#)
- static const [iostate](#) [eofbit](#)
- static const [iostate](#) [failbit](#)
- static const [fmtflags](#) [fixed](#)
- static const [fmtflags](#) [floatfield](#)
- static const [iostate](#) [goodbit](#)
- static const [fmtflags](#) [hex](#)
- static const [openmode](#) [in](#)
- static const [fmtflags](#) [internal](#)
- static const [fmtflags](#) [left](#)
- static const [fmtflags](#) [oct](#)
- static const [openmode](#) [out](#)
- static const [fmtflags](#) [right](#)
- static const [fmtflags](#) [scientific](#)
- static const [fmtflags](#) [showbase](#)
- static const [fmtflags](#) [showpoint](#)
- static const [fmtflags](#) [showpos](#)
- static const [fmtflags](#) [skipws](#)
- static const [openmode](#) [trunc](#)
- static const [fmtflags](#) [unitbuf](#)
- static const [fmtflags](#) [uppercase](#)

Protected Types

- enum { **[_S_local_word_size](#)** }

Protected Member Functions

- void **_M_cache_locale** (const [locale](#) &__loc)
- void **_M_call_callbacks** ([event](#) __ev) throw ()
- void **_M_dispose_callbacks** (void) throw ()
- [_Words](#) & **_M_grow_words** (int __index, bool __iword)
- void **_M_init** () throw ()
- template<typename _ValueT >
 [__ostream_type](#) & **_M_insert** (_ValueT __v)
- void **_M_move** ([ios_base](#) &) noexcept
- void **_M_swap** ([ios_base](#) &__rhs) noexcept
- void **init** ([basic_streambuf](#)< _CharT, _Traits > *__sb)
- void **move** ([basic_ios](#) &__rhs)
- void **move** ([basic_ios](#) &&__rhs)
- void **set_rdbuf** ([basic_streambuf](#)< _CharT, _Traits > *__sb)
- void **swap** ([basic_ostream](#) &__rhs)
- void **swap** ([basic_ios](#) &__rhs) noexcept

Protected Attributes

- [_Callback_list](#) * **_M_callbacks**
- const [__ctype_type](#) * **_M_ctype**
- [iostate](#) **_M_exception**
- char_type **_M_fill**
- bool **_M_fill_init**
- [fmtflags](#) **_M_flags**
- [locale](#) **_M_ios_locale**
- [_Words](#) **_M_local_word** [[_S_local_word_size](#)]
- const [__num_get_type](#) * **_M_num_get**
- const [__num_put_type](#) * **_M_num_put**
- [streamsize](#) **_M_precision**
- [basic_streambuf](#)< _CharT, _Traits > * **_M_streambuf**
- [iostate](#) **_M_streambuf_state**
- [basic_ostream](#)< _CharT, _Traits > * **_M_tie**
- [streamsize](#) **_M_width**
- [_Words](#) * **_M_word**
- int **_M_word_size**
- [_Words](#) **_M_word_zero**

5.618.1 Detailed Description

```
template<typename _CharT, typename _Traits>
class std::basic_ofstream< _CharT, _Traits >
```

Controlling output for files.

Template Parameters

<code>_CharT</code>	Type of character stream.
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .

This class supports reading from named files, using the inherited functions from `std::basic_ostream`. To control the associated sequence, an instance of `std::basic_filebuf` is used, which this page refers to as `sb`.

Definition at line 656 of file `fstream`.

5.618.2 Member Typedef Documentation

5.618.2.1 `template<typename _CharT, typename _Traits> typedef num_get<_CharT, istreambuf_iterator<_CharT, _Traits>> std::basic_ios<_CharT, _Traits>::__num_get_type [inherited]`

These are non-standard types.

Definition at line 91 of file `basic_ios.h`.

5.618.2.2 `typedef void(* std::ios_base::event_callback)(event __e, ios_base &__b, int __i) [inherited]`

The type of an event callback function.

Parameters

<code>__e</code>	One of the members of the event enum.
<code>__b</code>	Reference to the <code>ios_base</code> object.
<code>__i</code>	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several `ios_base` and `basic_ios` functions, specifically `imbue()`, `copyfmt()`, and `~ios()`.

Definition at line 504 of file `ios_base.h`.

5.618.2.3 `typedef _Ios_Fmtflags std::ios_base::fmtflags [inherited]`

This is a bitmask type.

`_Ios_Fmtflags` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `fmtflags` are:

- `boolalpha`

- `dec`
- `fixed`
- `hex`
- `internal`
- `left`
- `oct`
- `right`
- `scientific`
- `showbase`
- `showpoint`
- `showpos`
- `skipws`
- `unitbuf`
- `uppercase`
- `adjustfield`
- `basefield`
- `floatfield`

Definition at line 323 of file `ios_base.h`.

5.618.2.4 `typedef _ios_iostate std::ios_base::iostate` `[inherited]`

This is a bitmask type.

`_Ios_Iostate` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `iostate` are:

- `badbit`
- `eofbit`
- `failbit`
- `goodbit`

Definition at line 398 of file `ios_base.h`.

5.618.2.5 `typedef _Ios_Openmode std::ios_base::openmode` `[inherited]`

This is a bitmask type.

`_Ios_Openmode` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `openmode` are:

- `app`
- `ate`
- `binary`
- `in`
- `out`
- `trunc`

Definition at line 429 of file `ios_base.h`.

5.618.2.6 `typedef _Ios_Seekdir std::ios_base::seekdir` `[inherited]`

This is an enumerated type.

`_Ios_Seekdir` is implementation-defined. Defined values of type `seekdir` are:

- `beg`
- `cur`, equivalent to `SEEK_CUR` in the C standard library.
- `end`, equivalent to `SEEK_END` in the C standard library.

Definition at line 461 of file `ios_base.h`.

5.618.3 Member Enumeration Documentation**5.618.3.1** `enum std::ios_base::event` `[inherited]`

The set of events that may be passed to an event callback.

`erase_event` is used during `~ios()` and `copyfmt()`. `imbue_event` is used during `imbue()`. `copyfmt_event` is used during `copyfmt()`.

Definition at line 487 of file `ios_base.h`.

5.618.4 Constructor & Destructor Documentation**5.618.4.1** `template<typename _CharT, typename _Traits> std::basic_ofstream<_CharT, _Traits>::basic_ofstream ()` `[inline]`

Default constructor.

Initializes `sb` using its default constructor, and passes `&sb` to the base class initializer. Does not open any files (you haven't given it a filename to open).

Definition at line 682 of file `fstream`.

5.618.4.2 `template<typename _CharT, typename _Traits> std::basic_ofstream<_CharT, _Traits>::basic_ofstream (const char * __s, ios_base::openmode __mode = ios_base::out|ios_base::trunc)` `[inline]`, `[explicit]`

Create an output file stream.

Parameters

<code>__s</code>	Null terminated string specifying the filename.
<code>__mode</code>	Open file in specified mode (see std::ios_base).

`ios_base::out` | `ios_base::trunc` is automatically included in `__mode`.

Tip: When using `std::string` to hold the filename, you must use `.c_str()` before passing it to this constructor.

Definition at line 697 of file `fstream`.

5.618.4.3 `template<typename _CharT, typename _Traits> std::basic_ofstream< _CharT, _Traits >::basic_ofstream (const std::string & __s, ios_base::openmode __mode = ios_base::out|ios_base::trunc) [inline], [explicit]`

Create an output file stream.

Parameters

<code>__s</code>	<code>std::string</code> specifying the filename.
<code>__mode</code>	Open file in specified mode (see std::ios_base).

`ios_base::out` | `ios_base::trunc` is automatically included in `__mode`.

Definition at line 715 of file `fstream`.

5.618.4.4 `template<typename _CharT, typename _Traits> std::basic_ofstream< _CharT, _Traits >::~basic_ofstream () [inline]`

The destructor does nothing.

The file is closed by the filebuf object, not the formatting stream.

Definition at line 737 of file `fstream`.

5.618.5 Member Function Documentation

5.618.5.1 `const locale& std::ios_base::M_getloc () const [inline], [inherited]`

Locale access.

Returns

A reference to the current locale.

Like `getloc` above, but returns a reference instead of generating a copy.

Definition at line 774 of file `ios_base.h`.

Referenced by `std::time_get< _CharT, _Inlter >::do_date_order()`, `std::time_get< _CharT, _Inlter >::do_get()`, `std::money_get< _CharT, _Inlter >::do_get()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::time_get< _CharT, _Inlter >::do_get_date()`, `std::time_get< _CharT, _Inlter >::do_get_monthname()`, `std::time_get< _CharT, _Inlter >::do_get_time()`, `std::time_get< _CharT, _Inlter >::do_get_weekday()`, `std::time_get< _CharT, _Inlter >::do_get_year()`, `std::time_put< _CharT, _Outlter >::do_put()`, `std::num_put< _CharT, _Outlter >::do_put()`, `std::time_get< _CharT, _Inlter >::get()`, and `std::time_put< _CharT, _Outlter >::put()`.

5.618.5.2 `template<typename _CharT, typename _Traits> void std::basic_ostream< _CharT, _Traits >::_M_write (const char_type* __s, streamsize __n) [inline], [inherited]`

Core write functionality, without sentry.

Parameters

<code>__s</code>	The array to insert.
<code>__n</code>	Maximum number of characters to insert.

Definition at line 311 of file ostream.

Referenced by `std::basic_ostream< _CharT, _Traits >::write()`.

5.618.5.3 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::bad () const [inline], [inherited]`

Fast error checking.

Returns

True if the badbit is set.

Note that other iostate flags may also be set.

Definition at line 211 of file basic_ios.h.

5.618.5.4 `template<typename _CharT, typename _Traits > void std::basic_ios< _CharT, _Traits >::clear (iostate __state = goodbit) [inherited]`

[Re]sets the error state.

Parameters

<code>__state</code>	The new state flag(s) to set.
----------------------	-------------------------------

See `std::ios_base::iostate` for the possible bit values. Most users will not need to pass an argument.

Definition at line 41 of file basic_ios.tcc.

Referenced by `std::basic_ios< char, _Traits >::exceptions()`, `std::__detail::operator>>()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_ios< char, _Traits >::rdstate()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ios< char, _Traits >::setstate()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

5.618.5.5 `template<typename _CharT, typename _Traits > void std::basic_ofstream< _CharT, _Traits >::close ()`
`[inline]`

Close the file.

Calls `std::basic_filebuf::close()`. If that function fails, `failbit` is set in the stream's error state.

Definition at line 839 of file `fstream`.

5.618.5.6 `template<typename _CharT, typename _Traits > basic_ios< _CharT, _Traits > & std::basic_ios< _CharT, _Traits >::copyfmt (const basic_ios< _CharT, _Traits > & __rhs)` `[inherited]`

Copies fields of `__rhs` into this.

Parameters

<code>__rhs</code>	The source values for the copies.
--------------------	-----------------------------------

Returns

Reference to this object.

All fields of `__rhs` are copied into this object except that `rdbuf()` and `rdstate()` remain unchanged. All values in the `pword` and `iword` arrays are copied. Before copying, each callback is invoked with `erase_event`. After copying, each (new) callback is invoked with `copyfmt_event`. The final step is to copy `exceptions()`.

Definition at line 63 of file `basic_ios.tcc`.

References `std::basic_ios< _CharT, _Traits >::exceptions()`, `std::basic_ios< _CharT, _Traits >::fill()`, `std::ios_base::flags()`, `std::ios_base::getloc()`, `std::ios_base::precision()`, `std::basic_ios< _CharT, _Traits >::tie()`, `std::tie()`, and `std::ios_base::width()`.

Referenced by `std::basic_ios< char, _Traits >::rdbuf()`.

5.618.5.7 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::eof () const`
`[inline], [inherited]`

Fast error checking.

Returns

True if the `eofbit` is set.

Note that other `iostate` flags may also be set.

Definition at line 190 of file `basic_ios.h`.

5.618.5.8 `template<typename _CharT, typename _Traits> iostate std::basic_ios< _CharT, _Traits >::exceptions () const`
`[inline], [inherited]`

Throwing exceptions on errors.

Returns

The current exceptions mask.

This changes nothing in the stream. See the one-argument version of `exceptions(iostate)` for the meaning of the return value.

Definition at line 222 of file `basic_ios.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, and `std::basic_ios< char, _Traits >::setstate()`.

5.618.5.9 `template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits >::exceptions (iostate`
`__except) [inline], [inherited]`

Throwing exceptions on errors.

Parameters

<code>__except</code>	The new exceptions mask.
-----------------------	--------------------------

By default, error flags are set silently. You can set an exceptions mask for each stream; if a bit in the mask becomes set in the error flags, then an exception of type `std::ios_base::failure` is thrown.

If the error flag is already set when the exceptions mask is added, the exception is immediately thrown. Try running the following under GCC 3.1 or later:

```
1 #include <iostream>
2 #include <fstream>
3 #include <exception>
4
5 int main()
6 {
7     std::set_terminate (__gnu_cxx::__verbose_terminate_handler);
8
9     std::ifstream f ("/etc/motd");
10
11     std::cerr << "Setting badbit\n";
12     f.setstate (std::ios_base::badbit);
13
14     std::cerr << "Setting exception mask\n";
15     f.exceptions (std::ios_base::badbit);
16 }
```

Definition at line 257 of file `basic_ios.h`.

5.618.5.10 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::fail () const`
`[inline], [inherited]`

Fast error checking.

Returns

True if either the badbit or the failbit is set.

Checking the badbit in fail() is historical practice. Note that other iostate flags may also be set.

Definition at line 201 of file basic_ios.h.

Referenced by std::basic_ios< char, _Traits >::operator bool(), std::basic_ios< char, _Traits >::operator!(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ofstream< _CharT, _Traits >::seekp(), std::basic_istream< _CharT, _Traits >::tellg(), std::basic_ofstream< _CharT, _Traits >::tellp(), and std::regex_traits< _Ch_type >::value().

5.618.5.11 template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::fill () const
[inline], [inherited]

Retrieves the *empty* character.

Returns

The current fill character.

It defaults to a space (' ') in the current locale.

Definition at line 370 of file basic_ios.h.

Referenced by std::basic_ios< _CharT, _Traits >::copyfmt(), std::basic_ios< char, _Traits >::fill(), std::normal_distribution< _RealType >::operator()(), std::gamma_distribution< _RealType >::operator()(), std::negative_binomial_distribution< _IntType >::operator()(), and std::operator>>().

5.618.5.12 template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::fill (char_type __ch) [inline], [inherited]

Sets a new *empty* character.

Parameters

<code>__ch</code>	The new character.
-------------------	--------------------

Returns

The previous fill character.

The fill character is used to fill out space when P+ characters have been requested (e.g., via setw), Q characters are actually used, and Q<P. It defaults to a space (' ') in the current locale.

Definition at line 390 of file basic_ios.h.

5.618.5.13 `fmtflags std::ios_base::flags () const` `[inline],[inherited]`

Access to format flags.

Returns

The format control flags for both input and output.

Definition at line 619 of file `ios_base.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >::discard()`, `std::money_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::operator()()`, `std::discard_block_engine< _RandomNumberEngine, __p, __r >::operator()()`, `std::shuffle_order_engine< _RandomNumberEngine, __k >::operator()()`, `std::normal_distribution< _RealType >::operator()()`, `std::gamma_distribution< _RealType >::operator()()`, `std::binomial_distribution< _IntType >::operator()()`, `std::negative_binomial_distribution< _IntType >::operator()()`, `std::poisson_distribution< _IntType >::operator()()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, `std::__detail::operator>>()`, `std::operator>>()`, `std::linear_congruential_engine< _UIntType, __a, __c, __m >::seed()`, and `std::basic_istream< _CharT, _Traits >::sentry::sentry()`.

5.618.5.14 `fmtflags std::ios_base::flags (fmtflags __fmtfl)` `[inline],[inherited]`

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

Definition at line 630 of file `ios_base.h`.

5.618.5.15 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::flush ()` `[inherited]`

Synchronizing the stream buffer.

Returns

`*this`

If `rdbuf ()` is a null pointer, changes nothing.

Otherwise, calls `rdbuf ()->pubsync ()`, and if that returns `-1`, sets `badbit`.

Definition at line 211 of file `ostream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_ios< _CharT, _Traits >::setstate()`, and `std::basic_ostream< _CharT, _Traits >::tellp()`.

Referenced by `std::operator<<()`, and `std::basic_ostream< _CharT, _Traits >::write()`.

5.618.5.16 `locale std::ios_base::getloc () const` `[inline],[inherited]`

Locale access.

Returns

A copy of the current locale.

If `imbue(loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale()`, the global C++ locale.

Definition at line 763 of file `ios_base.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::money_put< _CharT, _Outlter >::do_put()`, `std::operator>>()`, and `std::ws()`.

5.618.5.17 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::good () const` `[inline],[inherited]`

Fast error checking.

Returns

True if no error flags are set.

A wrapper around `rdstate`.

Definition at line 180 of file `basic_ios.h`.

Referenced by `std::__detail::operator>>()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_istream< _CharT, _Traits >::sentry::sentry()`.

5.618.5.18 `template<typename _CharT, typename _Traits> locale std::basic_ios< _CharT, _Traits >::imbue (const locale & __loc)` `[inherited]`

Moves to a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls `ios_base::imbue(loc)`, and if a stream buffer is associated with this stream, calls that buffer's `pubimbue(loc)`.

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Definition at line 114 of file basic_ios.tcc.

References std::ios_base::imbue().

Referenced by std::basic_ios< char, _Traits >::fill().

5.618.5.19 `template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits >::init (basic_streambuf< _CharT, _Traits > * __sb) [protected], [inherited]`

All setup is performed here.

This is called from the public constructor. It is not virtual and cannot be redefined.

Definition at line 126 of file basic_ios.tcc.

Referenced by std::basic_ios< char, _Traits >::basic_ios().

5.618.5.20 `template<typename _CharT, typename _Traits > bool std::basic_ofstream< _CharT, _Traits >::is_open () [inline]`

Wrapper to test for an open file.

Returns

`rdbuf() -> is_open()`

Definition at line 778 of file fstream.

5.618.5.21 `long& std::ios_base::iword (int __ix) [inline], [inherited]`

Access to integer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to an integer associated with the index.

The `iword` function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 809 of file ios_base.h.

5.618.5.22 `template<typename _CharT, typename _Traits> char std::basic_ios<_CharT, _Traits>::narrow (char_type __c, char __default) const [inline], [inherited]`

Squeezes characters.

Parameters

<code>__c</code>	The character to narrow.
<code>__default</code>	The character to narrow.

Returns

The narrowed character.

Maps a character of `char_type` to a character of `char`, if possible.

Returns the result of

```
1 std::use_facet<ctype<char_type>> >(getloc()).narrow(c, default)
```

Additional l10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Definition at line 430 of file `basic_ios.h`.

5.618.5.23 `template<typename _CharT, typename _Traits> void std::basic_ofstream<_CharT, _Traits>::open (const char * __s, ios_base::openmode __mode = ios_base::out | ios_base::trunc) [inline]`

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Calls `std::basic_filebuf::open(__s, __mode|out|trunc)`. If that function fails, `failbit` is set in the stream's error state.

Tip: When using `std::string` to hold the filename, you must use `.c_str()` before passing it to this constructor.

Definition at line 799 of file `fstream`.

5.618.5.24 `template<typename _CharT, typename _Traits> void std::basic_ofstream<_CharT, _Traits>::open (const std::string & __s, ios_base::openmode __mode = ios_base::out | ios_base::trunc) [inline]`

Opens an external file.

Parameters

<code>__s</code>	The name of the file.
<code>__mode</code>	The open mode flags.

Calls `std::basic_filebuf::open(s,mode|out|trunc)`. If that function fails, `failbit` is set in the stream's error state.

Definition at line 820 of file `fstream`.

5.618.5.25 `template<typename _CharT, typename _Traits> std::basic_ios< _CharT, _Traits >::operator bool () const`
`[inline], [explicit], [inherited]`

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`

Definition at line 117 of file `basic_ios.h`.

5.618.5.26 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::operator! () const`
`[inline], [inherited]`

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`

Definition at line 125 of file `basic_ios.h`.

5.618.5.27 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits`
`>::operator<<(__ostream_type &)(__ostream_type &)__pf) [inline], [inherited]`

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `omanip` header.

Definition at line 108 of file `ostream`.

Referenced by `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, and `std::basic_ostream< ↵
 _CharT, _Traits >::sentry::sentry()`.

5.618.5.28 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits`
`>::operator<<(__ios_type &)(__ios_type &)__pf) [inline], [inherited]`

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `omanip` header.

Definition at line 117 of file `ostream`.

```
5.618.5.29 template<typename _CharT, typename _Traits> __ostream_type& std::basic_ofstream< _CharT, _Traits
>::operator<<( ios_base &(*)(ios_base &)_pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

Definition at line 127 of file `ostream`.

```
5.618.5.30 template<typename _CharT, typename _Traits> __ostream_type& std::basic_ofstream< _CharT, _Traits
>::operator<<( long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 166 of file `ostream`.

```
5.618.5.31 template<typename _CharT, typename _Traits> __ostream_type& std::basic_ofstream< _CharT, _Traits
>::operator<<( unsigned long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 170 of file `ostream`.

```
5.618.5.32 template<typename _CharT, typename _Traits> __ostream_type& std::basic_ofstream< _CharT, _Traits
>::operator<<( bool __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

<code>_↔</code>	A variable of builtin integral type.
<code>_n</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 174 of file `ostream`.

5.618.5.33 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::operator<<(short _n) [inherited]`

Integer arithmetic inserters.

Parameters

<code>_↔</code>	A variable of builtin integral type.
<code>_n</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 92 of file `ostream.tcc`.

References `std::ios_base::basefield`, `std::ios_base::flags()`, `std::ios_base::hex`, `std::ios_base::oct`, and `std::basic_↔ostream<_CharT, _Traits>::operator<<()`.

5.618.5.34 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(unsigned short _n) [inline], [inherited]`

Integer arithmetic inserters.

Parameters

<code>_↔</code>	A variable of builtin integral type.
<code>_n</code>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 181 of file ostream.

5.618.5.35 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::operator<<(int __n) [inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 106 of file ostream.tcc.

References `std::ios_base::basefield`, `std::ios_base::flags()`, `std::ios_base::hex`, `std::ios_base::oct`, and `std::basic_ostream< _CharT, _Traits>::operator<<()`.

5.618.5.36 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(unsigned int __n) [inline],[inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 192 of file ostream.

5.618.5.37 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(long long __n) [inline],[inherited]`

Integer arithmetic inserters.

Parameters

\leftrightarrow	A variable of builtin integral type.
$_n$	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 201 of file ostream.

```
5.618.5.38 template<typename _CharT, typename _Traits> __ostream_type& std::basic_ofstream<_CharT, _Traits>::operator<<( unsigned long long __n ) [inline],[inherited]
```

Integer arithmetic inserters.

Parameters

\leftrightarrow	A variable of builtin integral type.
$_n$	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 205 of file ostream.

```
5.618.5.39 template<typename _CharT, typename _Traits> __ostream_type& std::basic_ofstream<_CharT, _Traits>::operator<<( double __f ) [inline],[inherited]
```

Floating point arithmetic inserters.

Parameters

\leftrightarrow	A variable of builtin floating point type.
$_ \leftrightarrow$	
$_ \leftrightarrow$	
$_ \leftrightarrow$	
f	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 220 of file `ostream`.

5.618.5.40 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(float __f) [inline],[inherited]`

Floating point arithmetic inserters.

Parameters

↩	A variable of builtin floating point type.
↩	
↩	
↩	
<i>f</i>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 224 of file `ostream`.

5.618.5.41 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(long double __f) [inline],[inherited]`

Floating point arithmetic inserters.

Parameters

↩	A variable of builtin floating point type.
↩	
↩	
↩	
<i>f</i>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 232 of file `ostream`.

5.618.5.42 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(const void *__p) [inline],[inherited]`

Pointer arithmetic inserters.

Parameters

<code>_↔ _p</code>	A variable of pointer type.
------------------------	-----------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 245 of file `ostream`.

5.618.5.43 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::operator<<(__streambuf_type * __sb) [inherited]`

Extracting from another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is `NULL`, the stream will set `failbit` in its error state.

Characters are extracted from `__sb` and inserted into `*this` until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output sequence fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs while getting a character from `__sb`, which sets `failbit` in the error state

If the function inserts no characters, `failbit` is set.

Definition at line 120 of file `ostream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ostream< _CharT, _Traits>::put()`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

5.618.5.44 `streamsize std::ios_base::precision () const` `[inline],[inherited]`

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Definition at line 689 of file `ios_base.h`.

Referenced by `std::basic_ios<_CharT,_Traits>::copyfmt()`, `std::num_get<_CharT,_InIter>::do_get()`, `std::normal_distribution<_RealType>::operator()()`, `std::gamma_distribution<_RealType>::operator()()`, `std::negative_binomial_distribution<_IntType>::operator()()`, and `std::operator>>()`.

5.618.5.45 `streamsize std::ios_base::precision (streamsize __prec)` `[inline],[inherited]`

Changing flags.

Parameters

<code>__prec</code>	The new precision value.
---------------------	--------------------------

Returns

The previous value of `precision()`.

Definition at line 698 of file `ios_base.h`.

5.618.5.46 `template<typename _CharT,typename _Traits> basic_ostream<_CharT,_Traits> & std::basic_ostream<_CharT,_Traits>::put (char_type __c)` `[inherited]`

Simple insertion.

Parameters

<code>__c</code>	The character to insert.
------------------	--------------------------

Returns

`*this`

Tries to insert `__c`.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 149 of file ostream.tcc.

References std::ios_base::badbit, std::ios_base::goodbit, std::basic_ios< _CharT, _Traits >::rdbuf(), std::basic_ios< _CharT, _Traits >::setstate(), and std::basic_ofstream< _CharT, _Traits >::write().

Referenced by std::basic_ofstream< _CharT, _Traits >::operator<<().

5.618.5.47 void*& std::ios_base::pword (int __ix) [inline],[inherited]

Access to void pointer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to a void* associated with the index.

The pword function provides access to an array of pointers that can be used for any purpose. The array grows as required to hold the supplied index. All pointers in the array are initialized to 0.

The implementation reserves several indices. You should use xalloc to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 830 of file ios_base.h.

5.618.5.48 template<typename _CharT, typename _Traits> basic_streambuf< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::rdbuf (basic_streambuf< _CharT, _Traits > * __sb) [inherited]

Changing the underlying buffer.

Parameters

<code>__sb</code>	The new stream buffer.
-------------------	------------------------

Returns

The previous stream buffer.

Associates a new buffer with the current stream, and clears the error state.

Due to historical accidents which the LWG refuses to correct, the I/O library suffers from a design error: this function is hidden in derived classes by overrides of the zero-argument rdbuf(), which is non-virtual for hysterical raisins. As a result, you must use explicit qualifications to access this function via any derived class. For example:

```

1 std::fstream      foo;           // or some other derived type
2 std::streambuf*   p = .....;
3
4 foo.ios::rdbuf(p);             // ios == basic_ios<char>

```

Definition at line 53 of file basic_ios.tcc.

5.618.5.49 `template<typename _CharT, typename _Traits > __filebuf_type* std::basic_ofstream< _CharT, _Traits >::rdbuf () const [inline]`

Accessing the underlying buffer.

Returns

The current basic_filebuf buffer.

This hides both signatures of std::basic_ios::rdbuf().

Definition at line 770 of file fstream.

5.618.5.50 `template<typename _CharT, typename _Traits> iostate std::basic_ios< _CharT, _Traits >::rdstate () const [inline], [inherited]`

Returns the error state of the stream buffer.

Returns

A bit pattern (well, isn't everything?)

See std::ios_base::iostate for the possible bit values. Most users will call one of the interpreting wrappers, e.g., good().

Definition at line 137 of file basic_ios.h.

Referenced by std::basic_ios< char, _Traits >::bad(), std::basic_ios< char, _Traits >::eof(), std::basic_ios< char, _Traits >::fail(), std::basic_ios< char, _Traits >::good(), std::basic_istream< _CharT, _Traits >::putback(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ios< char, _Traits >::setstate(), and std::basic_istream< _CharT, _Traits >::unget().

5.618.5.51 `void std::ios_base::register_callback (event_callback __fn, int __index) [inherited]`

Add the callback __fn with parameter __index.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple

copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

5.618.5.52 `template<typename _CharT, typename _Traits > basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::seekp (pos_type __pos) [inherited]`

Changing the current write position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

*this

If `fail()` is not true, calls `rdbuf() -> pubseekpos(pos)`. If that function fails, sets failbit.

Definition at line 258 of file ostream.tcc.

References `std::ios_base::badbit`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::ios_base::out`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

Referenced by `std::basic_ostream< _CharT, _Traits >::tellp()`.

5.618.5.53 `template<typename _CharT, typename _Traits > basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::seekp (off_type __off, ios_base::seekdir __dir) [inherited]`

Changing the current write position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

*this

If `fail()` is not true, calls `rdbuf() -> pubseekoff(off, dir)`. If that function fails, sets failbit.

Definition at line 290 of file ostream.tcc.

References `std::ios_base::badbit`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::ios_base::out`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

5.618.5.54 `fmtflags std::ios_base::setf (fmtflags __fmtfl) [inline], [inherited]`

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
----------------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set.

Definition at line 646 of file `ios_base.h`.

Referenced by `std::boolalpha()`, `std::dec()`, `std::fixed()`, `std::hex()`, `std::hexfloat()`, `std::left()`, `std::oct()`, `std::__detail<::operator>>()`, `std::right()`, `std::scientific()`, `std::showbase()`, `std::showpoint()`, `std::showpos()`, `std::skipws()`, `std::unitbuf()`, and `std::uppercase()`.

5.618.5.55 `fmtflags std::ios_base::setf (fmtflags __fmtfl, fmtflags __mask)` `[inline],[inherited]`

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
<code>__mask</code>	The flags mask for <code>__fmtfl</code> .

Returns

The previous format control flags.

This function clears `mask` in the format flags, then sets `__fmtfl` & `mask`. An example mask is `ios_base::adjustfield`.

Definition at line 663 of file `ios_base.h`.

5.618.5.56 `template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits >::setstate (iostate __state)` `[inline],[inherited]`

Sets additional flags in the error state.

Parameters

<code>__state</code>	The additional state flag(s) to set.
----------------------	--------------------------------------

See `std::ios_base::iostate` for the possible bit values.

Definition at line 157 of file `basic_ios.h`.

Referenced by std::basic_ostream< _CharT, _Traits >::flush(), std::basic_istream< _CharT, _Traits >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::getline(), std::basic_istream< _CharT, _Traits >::ignore(), std::basic_ostream< _CharT, _Traits >::operator<<(), std::operator<<(), std::basic_istream< _CharT, _Traits >::operator>>(), std::tr2::operator>>(), std::operator>>(), std::basic_istream< _CharT, _Traits >::peek(), std::basic_ostream< _CharT, _Traits >::put(), std::basic_istream< _CharT, _Traits >::putback(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::readsome(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_ostream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::sync(), std::basic_istream< _CharT, _Traits >::unset(), and std::ws().

5.618.5.57 static bool std::ios_base::sync_with_stdio (bool __sync = true) [static],[inherited]

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., stdout) and the standard C++ objects (e.g., cout). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstream.html#std.io.filestreams.binary>

5.618.5.58 template<typename _CharT, typename _Traits > basic_ostream< _CharT, _Traits >::pos_type
std::basic_ostream< _CharT, _Traits >::tellp () [inherited]

Getting the current write position.

Returns

A file position object.

If fail() is not false, returns pos_type(-1) to indicate failure. Otherwise returns rdbuf() ->pubseekoff(0, cur, out).

Definition at line 237 of file ostream.tcc.

References std::ios_base::badbit, std::ios_base::cur, std::basic_ios< _CharT, _Traits >::fail(), std::ios_base::out, std::basic_ios< _CharT, _Traits >::rdbuf(), and std::basic_ostream< _CharT, _Traits >::seekp().

Referenced by std::basic_ostream< _CharT, _Traits >::flush().


```
5.618.5.59  template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>* std::basic_ios<_CharT,
    _Traits>::tie( ) const    [inline], [inherited]
```

Fetches the current *tied* stream.

Returns

A pointer to the tied stream, or NULL if the stream is not tied.

A stream may be *tied* (or synchronized) to a second output stream. When this stream performs any I/O, the tied stream is first flushed. For example, `std::cin` is tied to `std::cout`.

Definition at line 295 of file `basic_ios.h`.

Referenced by `std::basic_ios<char, _Traits>::basic_ios()`, `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`, and `std::basic_istream<_CharT, _Traits>::sentry::sentry()`.

```
5.618.5.60  template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>* std::basic_ios<_CharT,
    _Traits>::tie( basic_ostream<_CharT, _Traits>* __tiestr )    [inline], [inherited]
```

Ties this stream to an output stream.

Parameters

<code>__tiestr</code>	The output stream.
-----------------------	--------------------

Returns

The previously tied output stream, or NULL if the stream was not tied.

This sets up a new tie; see `tie()` for more.

Definition at line 307 of file `basic_ios.h`.

```
5.618.5.61  void std::ios_base::unsetf( fmtflags __mask )    [inline], [inherited]
```

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

Definition at line 678 of file `ios_base.h`.

Referenced by `std::defaultfloat()`, `std::noboolalpha()`, `std::noshowbase()`, `std::noshowpoint()`, `std::noshowpos()`, `std::noskipws()`, `std::nounitbuf()`, and `std::nouppercase()`.

5.618.5.62 `template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::widen (char __c)`
`const [inline],[inherited]`

Widens characters.

Parameters

<code>__c</code>	The character to widen.
------------------	-------------------------

Returns

The widened character.

Maps a character of `char` to a character of `char_type`.

Returns the result of

```
1 std::use_facet<ctype<char_type>> >(getloc()).widen(c)
```

Additional l10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Definition at line 449 of file `basic_ios.h`.

Referenced by `std::basic_ios< char, _Traits >::fill()`, `std::getline()`, `std::normal_distribution< _RealType >::operator()()`, `std::gamma_distribution< _RealType >::operator()()`, `std::negative_binomial_distribution< _IntType >::operator()()`, `std::tr2::operator>>()`, and `std::operator>>()`.

5.618.5.63 `streamsize std::ios_base::width () const [inline],[inherited]`

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Definition at line 712 of file `ios_base.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::money_get< _CharT, _Inlter >::do_get()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::num_put< _CharT, _Outlter >::do_put()`, and `std::operator>>()`.

5.618.5.64 `streamsize std::ios_base::width (streamsize __wide) [inline],[inherited]`

Changing flags.

Parameters

<code>__wide</code>	The new width value.
---------------------	----------------------

Returns

The previous value of `width()`.

Definition at line 721 of file `ios_base.h`.

5.618.5.65 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::write (const char_type * __s, streamsize __n) [inherited]`

Character string insertion.

Parameters

<code>__s</code>	The array to insert.
<code>__n</code>	Maximum number of characters to insert.

Returns

`*this`

Characters are copied from `__s` and inserted into the stream until one of the following happens:

- `__n` characters are inserted
- inserting into the output sequence fails (in this case, `badbit` will be set in the stream's error state)

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 183 of file `ostream.tcc`.

References `std::basic_ostream< _CharT, _Traits>::_M_write()`, `std::ios_base::badbit`, and `std::basic_ostream< _CharT, _Traits>::flush()`.

Referenced by `std::basic_ostream< _CharT, _Traits>::put()`.

5.618.5.66 static int std::ios_base::xalloc () throw [static],[inherited]

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the iword and pword functions. The expectation is that an application calls xalloc in order to obtain an index in the iword and pword arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. xalloc is guaranteed to return an index that is safe to use in the iword and pword arrays.

5.618.6 Member Data Documentation

5.618.6.1 const fmtflags std::ios_base::adjustfield [static],[inherited]

A mask of left|right|internal. Useful for the 2-arg form of setf.

Definition at line 378 of file ios_base.h.

Referenced by std::money_get< _CharT, _InIter >::do_get(), std::num_put< _CharT, _OutIter >::do_put(), std::internal(), std::left(), and std::right().

5.618.6.2 const openmode std::ios_base::app [static],[inherited]

Seek to end before each write.

Definition at line 432 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::overflow(), and std::basic_filebuf< _CharT, _Traits >::xsputn().

5.618.6.3 const openmode std::ios_base::ate [static],[inherited]

Open and seek to end immediately after opening.

Definition at line 435 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::open().

5.618.6.4 `const iostate std::ios_base::badbit` `[static], [inherited]`

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Definition at line 402 of file `ios_base.h`.

Referenced by `std::basic_ios< char, _Traits >::bad()`, `std::basic_ios< char, _Traits >::fail()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::basic_ostream< _CharT, _Traits >::write()`.

5.618.6.5 `const fmtflags std::ios_base::basefield` `[static], [inherited]`

A mask of `dec|oct|hex`. Useful for the 2-arg form of `setf`.

Definition at line 381 of file `ios_base.h`.

Referenced by `std::dec()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::hex()`, `std::oct()`, and `std::basic_ostream< _CharT, _Traits >::operator<<()`.

5.618.6.6 `const seekdir std::ios_base::beg` `[static], [inherited]`

Request a seek relative to the beginning of the stream.

Definition at line 464 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::seekpos()`, and `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync()`.

5.618.6.7 `const openmode std::ios_base::binary` `[static], [inherited]`

Perform input and output in binary mode (as opposed to text mode). This is probably not what you think it is; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>.

Definition at line 440 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::showmanyc()`.

5.618.6.8 `const fmtflags std::ios_base::boolalpha` `[static], [inherited]`

Insert/extract `bool` in alphabetic rather than numeric format.

Definition at line 326 of file `ios_base.h`.

Referenced by `std::boolalpha()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, and `std::noboolalpha()`.

5.618.6.9 const seekdir std::ios_base::cur [static], [inherited]

Request a seek relative to the current position within the sequence.

Definition at line 467 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::imbue(), std::basic_filebuf< _CharT, _Traits >::overflow(), std::basic_filebuf< _CharT, _Traits >::pbackfail(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff(), std::basic_filebuf< _CharT, _Traits >::seekoff(), __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync(), std::basic_istream< _CharT, _Traits >::tellg(), and std::basic_ostream< _CharT, _Traits >::tellp().

5.618.6.10 const fmtflags std::ios_base::dec [static], [inherited]

Converts integer input or generates integer output in decimal base.

Definition at line 329 of file ios_base.h.

Referenced by std::dec().

5.618.6.11 const seekdir std::ios_base::end [static], [inherited]

Request a seek relative to the current end of the sequence.

Definition at line 470 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::open(), and std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff().

5.618.6.12 const iostate std::ios_base::eofbit [static], [inherited]

Indicates that an input operation reached the end of an input sequence.

Definition at line 405 of file ios_base.h.

Referenced by std::time_get< _CharT, _Inlter >::do_get(), std::num_get< _CharT, _Inlter >::do_get(), std::time_get< _CharT, _Inlter >::do_get_date(), std::time_get< _CharT, _Inlter >::do_get_monthname(), std::time_get< _CharT, _Inlter >::do_get_time(), std::time_get< _CharT, _Inlter >::do_get_weekday(), std::time_get< _CharT, _Inlter >::do_get_year(), std::basic_ios< char, _Traits >::eof(), std::basic_istream< _CharT, _Traits >::get(), std::time_get< _CharT, _Inlter >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::basic_istream< _CharT, _Traits >::ignore(), std::basic_istream< _CharT, _Traits >::operator>>(), std::operator>>(), std::basic_istream< _CharT, _Traits >::peek(), std::basic_istream< _CharT, _Traits >::putback(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::readsome(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_istream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::unget(), and std::ws().

5.618.6.13 const iostate std::ios_base::failbit [static], [inherited]

Indicates that an input operation failed to read the expected characters, or that an output operation failed to generate the desired characters.

Definition at line 410 of file ios_base.h.

Referenced by std::time_get< _CharT, _Inlter >::do_date_order(), std::num_get< _CharT, _Inlter >::do_get(), std::time_get< _CharT, _Inlter >::do_get_monthname(), std::time_get< _CharT, _Inlter >::do_get_weekday(), std::time_get< _CharT, _Inlter >::do_get_year(), std::basic_ios< char, _Traits >::fail(), std::basic_istream< _CharT, _Traits >::get(), std::time_get< _CharT, _Inlter >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::basic_ostream< _CharT, _Traits >::operator<<(), std::basic_istream< _CharT, _Traits >::operator>>(), std::operator>>(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_ostream< _CharT, _Traits >::sentry::sentry(), and std::basic_istream< _CharT, _Traits >::sentry::sentry().

5.618.6.14 `const fmtflags std::ios_base::fixed` `[static], [inherited]`

Generate floating-point output in fixed-point notation.

Definition at line 332 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _Inlter >::do_get()`, `std::fixed()`, and `std::hexfloat()`.

5.618.6.15 `const fmtflags std::ios_base::floatfield` `[static], [inherited]`

A mask of scientific|fixed. Useful for the 2-arg form of `setf`.

Definition at line 384 of file `ios_base.h`.

Referenced by `std::defaultfloat()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::fixed()`, `std::hexfloat()`, and `std::scientific()`.

5.618.6.16 `const iostate std::ios_base::goodbit` `[static], [inherited]`

Indicates all is well.

Definition at line 413 of file `ios_base.h`.

Referenced by `std::time_get< _CharT, _Inlter >::do_date_order()`, `std::time_get< _CharT, _Inlter >::do_get()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::time_get< _CharT, _Inlter >::do_get_monthname()`, `std::time_get< _CharT, _Inlter >::do_get_weekday()`, `std::time_get< _CharT, _Inlter >::do_get_year()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::time_get< _CharT, _Inlter >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_ios< char, _Traits >::rdstate()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sync()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

5.618.6.17 `const fmtflags std::ios_base::hex` `[static], [inherited]`

Converts integer input or generates integer output in hexadecimal base.

Definition at line 335 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _Inlter >::do_get()`, `std::num_put< _CharT, _Outlter >::do_put()`, `std::hex()`, and `std::basic_ostream< _CharT, _Traits >::operator<<()`.

5.618.6.18 `const openmode std::ios_base::in` `[static], [inherited]`

Open for input. Default for `ifstream` and `fstream`.

Definition at line 443 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos()`, `std::basic_filebuf< _CharT, _Traits >::showmanyc()`, `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::underflow()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

5.618.6.19 `const fmtflags std::ios_base::internal` `[static], [inherited]`

Adds fill characters at a designated internal point in certain generated output, or identical to `right` if no such point is designated.

Definition at line 340 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, and `std::ios_base::internal()`.

5.618.6.20 `const fmtflags std::ios_base::left` `[static], [inherited]`

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.)

Definition at line 344 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, and `std::ios_base::left()`.

5.618.6.21 `const fmtflags std::ios_base::oct` `[static], [inherited]`

Converts integer input or generates integer output in octal base.

Definition at line 347 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _InIter >::do_get()`, `std::oct()`, and `std::basic_ostream< _CharT, _Traits >::operator<<()`.

5.618.6.22 `const openmode std::ios_base::out` `[static], [inherited]`

Open for output. Default for `ofstream` and `fstream`.

Definition at line 446 of file `ios_base.h`.

Referenced by `std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow()`, `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::pbackfail()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos()`, `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

5.618.6.23 `const fmtflags std::ios_base::right` `[static], [inherited]`

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Definition at line 351 of file `ios_base.h`.

Referenced by `std::right()`.

5.618.6.24 `const fmtflags std::ios_base::scientific` `[static], [inherited]`

Generates floating-point output in scientific notation.

Definition at line 354 of file `ios_base.h`.

Referenced by `std::hexfloat()`, and `std::scientific()`.

5.618.6.25 `const fmtflags std::ios_base::showbase` `[static], [inherited]`

Generates a prefix indicating the numeric base of generated integer output.

Definition at line 358 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::noshowbase()`, and `std::showbase()`.

5.618.6.26 `const fmtflags std::ios_base::showpoint` `[static], [inherited]`

Generates a decimal-point character unconditionally in generated floating-point output.

Definition at line 362 of file `ios_base.h`.

Referenced by `std::noshowpoint()`, and `std::showpoint()`.

5.618.6.27 `const fmtflags std::ios_base::showpos` `[static], [inherited]`

Generates a + sign in non-negative generated numeric output.

Definition at line 365 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _InIter >::do_get()`, `std::noshowpos()`, and `std::showpos()`.

5.618.6.28 `const fmtflags std::ios_base::skipws` `[static], [inherited]`

Skips leading white space before certain input operations.

Definition at line 368 of file `ios_base.h`.

Referenced by `std::noskipws()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, and `std::skipws()`.

5.618.6.29 `const openmode std::ios_base::trunc` `[static], [inherited]`

Open for input. Default for `ofstream`.

Definition at line 449 of file `ios_base.h`.

5.618.6.30 `const fmtflags std::ios_base::unitbuf` `[static], [inherited]`

Flushes output after each output operation.

Definition at line 371 of file `ios_base.h`.

Referenced by `std::noinitbuf()`, and `std::unitbuf()`.

5.618.6.31 `const fmtflags std::ios_base::uppercase` `[static], [inherited]`

Replaces certain lowercase letters with their uppercase equivalents in generated output.

Definition at line 375 of file `ios_base.h`.

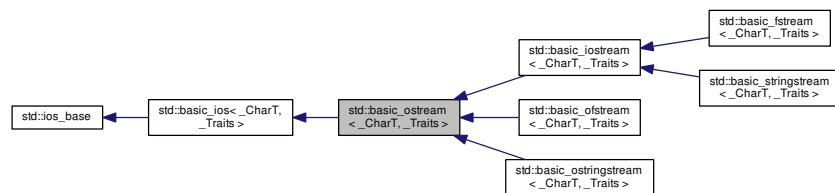
Referenced by `std::num_get<_CharT, _InIter>::do_get()`, `std::num_put<_CharT, _OutIter>::do_put()`, `std::nouppercase()`, and `std::uppercase()`.

The documentation for this class was generated from the following file:

- [fstream](#)

5.619 std::basic_ostream<_CharT, _Traits> Class Template Reference

Inheritance diagram for `std::basic_ostream<_CharT, _Traits>`:



Classes

- class [sentry](#)

Public Types

- typedef [ctype](#)< _CharT > **__ctype_type**
 - typedef [basic_ios](#)< _CharT, _Traits > **__ios_type**
 - typedef [num_put](#)< _CharT, [ostreambuf_iterator](#)< _CharT, _Traits > > **__num_put_type**
 - typedef [basic_ostream](#)< _CharT, _Traits > **__ostream_type**
 - typedef [basic_streambuf](#)< _CharT, _Traits > **__streambuf_type**
 - typedef _CharT **char_type**
 - enum [event](#) { [erase_event](#), [imbue_event](#), [copyfmt_event](#) }
 - typedef void(* [event_callback](#)) (event __e, [ios_base](#) &__b, int __i)
 - typedef _ios_Fmtflags [fmtflags](#)
 - typedef _Traits::int_type **int_type**
 - typedef int **io_state**
 - typedef _ios_iostate [iostate](#)
 - typedef _Traits::off_type **off_type**
 - typedef int **open_mode**
 - typedef _ios_Openmode [openmode](#)
 - typedef _Traits::pos_type **pos_type**
 - typedef int **seek_dir**
 - typedef _ios_Seekdir [seekdir](#)
 - typedef [std::streamoff](#) **streamoff**
 - typedef [std::streampos](#) **streampos**
 - typedef _Traits **traits_type**
-
- typedef [num_get](#)< _CharT, [istreambuf_iterator](#)< _CharT, _Traits > > **__num_get_type**

Public Member Functions

- [basic_ostream](#) ([__streambuf_type](#) *__sb)
- virtual [~basic_ostream](#) ()
- const [locale](#) & [_M_getloc](#) () const
- template<typename _ValueT >
 [basic_ostream](#)< _CharT, _Traits > & [_M_insert](#) (_ValueT __v)
- void [_M_setstate](#) ([iostate](#) __state)
- bool [bad](#) () const
- void [clear](#) ([iostate](#) __state=[goodbit](#))
- [basic_ios](#) & [copyfmt](#) (const [basic_ios](#) &__rhs)
- bool [eof](#) () const
- [iostate](#) [exceptions](#) () const
- void [exceptions](#) ([iostate](#) __except)
- bool [fail](#) () const
- [char_type](#) [fill](#) () const
- [char_type](#) [fill](#) ([char_type](#) __ch)
- [fmtflags](#) [flags](#) () const
- [fmtflags](#) [flags](#) ([fmtflags](#) __fmtfl)
- [__ostream_type](#) & [flush](#) ()
- [locale](#) [getloc](#) () const
- bool [good](#) () const
- [locale](#) [imbue](#) (const [locale](#) &__loc)

- long & [iword](#) (int __ix)
 - char [narrow](#) (char_type __c, char __dfault) const
 - [__ostream_type](#) & [operator<<](#) (const void *__p)
 - [__ostream_type](#) & [operator<<](#) ([__streambuf_type](#) *__sb)
 - [streamsize](#) [precision](#) () const
 - [streamsize](#) [precision](#) ([streamsize](#) __prec)
 - void *& [pword](#) (int __ix)
 - [basic_streambuf](#)< _CharT, _Traits > * [rdbuf](#) () const
 - [basic_streambuf](#)< _CharT, _Traits > * [rdbuf](#) ([basic_streambuf](#)< _CharT, _Traits > *__sb)
 - [iostate](#) [rdstate](#) () const
 - void [register_callback](#) ([event_callback](#) __fn, int __index)
 - [__ostream_type](#) & [seekp](#) (pos_type)
 - [__ostream_type](#) & [seekp](#) (off_type, [ios_base::seekdir](#))
 - [fmtflags](#) [setf](#) ([fmtflags](#) __fmtfl)
 - [fmtflags](#) [setf](#) ([fmtflags](#) __fmtfl, [fmtflags](#) __mask)
 - void [setstate](#) ([iostate](#) __state)
 - pos_type [tellp](#) ()
 - [basic_ostream](#)< _CharT, _Traits > * [tie](#) () const
 - [basic_ostream](#)< _CharT, _Traits > * [tie](#) ([basic_ostream](#)< _CharT, _Traits > *__tiestr)
 - void [unsetf](#) ([fmtflags](#) __mask)
 - char_type [widen](#) (char __c) const
 - [streamsize](#) [width](#) () const
 - [streamsize](#) [width](#) ([streamsize](#) __wide)
-
- [__ostream_type](#) & [operator<<](#) ([__ostream_type](#) &(*__pf)([__ostream_type](#) &))
 - [__ostream_type](#) & [operator<<](#) ([__ios_type](#) &(*__pf)([__ios_type](#) &))
 - [__ostream_type](#) & [operator<<](#) ([ios_base](#) &(*__pf)([ios_base](#) &))

Inserters

All the `operator<<` functions (aka formatted output functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This can have several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- [__ostream_type](#) & [operator<<](#) (long __n)
 - [__ostream_type](#) & [operator<<](#) (unsigned long __n)
 - [__ostream_type](#) & [operator<<](#) (bool __n)
 - [__ostream_type](#) & [operator<<](#) (short __n)
 - [__ostream_type](#) & [operator<<](#) (unsigned short __n)
 - [__ostream_type](#) & [operator<<](#) (int __n)
 - [__ostream_type](#) & [operator<<](#) (unsigned int __n)
 - [__ostream_type](#) & [operator<<](#) (long long __n)
 - [__ostream_type](#) & [operator<<](#) (unsigned long long __n)
-
- [__ostream_type](#) & [operator<<](#) (double __f)
 - [__ostream_type](#) & [operator<<](#) (float __f)
 - [__ostream_type](#) & [operator<<](#) (long double __f)

Unformatted Output Functions

All the unformatted output functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This has several effects, concluding with the setting of a status flag; see the `sentry` documentation for more.

If the `sentry` status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state. If `badbit` is on in the stream's exceptions mask, the exception will be rethrown without completing its actions.

- `__ostream_type & put (char_type __c)`
- `void _M_write (const char_type *__s, streamsize __n)`
- `__ostream_type & write (const char_type *__s, streamsize __n)`
- `operator bool () const`
- `bool operator! () const`

Static Public Member Functions

- `static bool sync_with_stdio (bool __sync=true)`
- `static int xalloc () throw ()`

Static Public Attributes

- `static const fmtflags adjustfield`
- `static const openmode app`
- `static const openmode ate`
- `static const iostate badbit`
- `static const fmtflags basefield`
- `static const seekdir beg`
- `static const openmode binary`
- `static const fmtflags boolalpha`
- `static const seekdir cur`
- `static const fmtflags dec`
- `static const seekdir end`
- `static const iostate eofbit`
- `static const iostate failbit`
- `static const fmtflags fixed`
- `static const fmtflags floatfield`
- `static const iostate goodbit`
- `static const fmtflags hex`
- `static const openmode in`
- `static const fmtflags internal`
- `static const fmtflags left`
- `static const fmtflags oct`
- `static const openmode out`
- `static const fmtflags right`
- `static const fmtflags scientific`
- `static const fmtflags showbase`
- `static const fmtflags showpoint`
- `static const fmtflags showpos`
- `static const fmtflags skipws`
- `static const openmode trunc`
- `static const fmtflags unitbuf`
- `static const fmtflags uppercase`

Protected Types

- enum { **_S_local_word_size** }

Protected Member Functions

- **basic_ostream** ([basic_ostream](#)<_CharT, _Traits> &)
- **basic_ostream** (const [basic_ostream](#) &)=delete
- **basic_ostream** ([basic_ostream](#) &&__rhs)
- void **_M_cache_locale** (const [locale](#) &__loc)
- void **_M_call_callbacks** ([event](#) __ev) throw ()
- void **_M_dispose_callbacks** (void) throw ()
- [_Words](#) & **_M_grow_words** (int __index, bool __iword)
- void **_M_init** () throw ()
- template<typename _ValueT>
 [__ostream_type](#) & **_M_insert** (_ValueT __v)
- void **_M_move** ([ios_base](#) &) noexcept
- void **_M_swap** ([ios_base](#) &__rhs) noexcept
- void **init** ([basic_streambuf](#)<_CharT, _Traits> *__sb)
- void **move** ([basic_ios](#) &__rhs)
- void **move** ([basic_ios](#) &&__rhs)
- [basic_ostream](#) & **operator=** (const [basic_ostream](#) &)=delete
- [basic_ostream](#) & **operator=** ([basic_ostream](#) &&__rhs)
- void **set_rdbuf** ([basic_streambuf](#)<_CharT, _Traits> *__sb)
- void **swap** ([basic_ostream](#) &__rhs)
- void **swap** ([basic_ios](#) &__rhs) noexcept

Protected Attributes

- [_Callback_list](#) * **_M_callbacks**
- const [__ctype_type](#) * **_M_ctype**
- [iostate](#) **_M_exception**
- [char_type](#) **_M_fill**
- bool **_M_fill_init**
- [fmtflags](#) **_M_flags**
- [locale](#) **_M_ios_locale**
- [_Words](#) **_M_local_word** [[_S_local_word_size](#)]
- const [__num_get_type](#) * **_M_num_get**
- const [__num_put_type](#) * **_M_num_put**
- [streamsize](#) **_M_precision**
- [basic_streambuf](#)<_CharT, _Traits> * **_M_streambuf**
- [iostate](#) **_M_streambuf_state**
- [basic_ostream](#)<_CharT, _Traits> * **_M_tie**
- [streamsize](#) **_M_width**
- [_Words](#) * **_M_word**
- int **_M_word_size**
- [_Words](#) **_M_word_zero**

Friends

- class **sentry**

5.619.1 Detailed Description

```
template<typename _CharT, typename _Traits>
class std::basic_ostream< _CharT, _Traits >
```

Template class basic_ostream.

Template Parameters

<code>_CharT</code>	Type of character stream.
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .

This is the base class for all output streams. It provides text formatting of all builtin types, and communicates with any class derived from `basic_streambuf` to do the actual output.

Definition at line 86 of file `iosfwd`.

5.619.2 Member Typedef Documentation

5.619.2.1 `template<typename _CharT, typename _Traits> typedef num_get<_CharT, istreambuf_iterator<_CharT, _Traits>
> std::basic_ios< _CharT, _Traits >::__num_get_type` `[inherited]`

These are non-standard types.

Definition at line 91 of file `basic_ios.h`.

5.619.2.2 `typedef void(* std::ios_base::event_callback) (event __e, ios_base &__b, int __i)` `[inherited]`

The type of an event callback function.

Parameters

<code>↔ __e</code>	One of the members of the event enum.
<code>↔ __b</code>	Reference to the <code>ios_base</code> object.
<code>↔ __i</code>	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several `ios_base` and `basic_ios` functions, specifically `imbue()`, `copyfmt()`, and `~ios()`.

Definition at line 504 of file `ios_base.h`.

5.619.2.3 `typedef _Ios_Fmtflags std::ios_base::fmtflags` `[inherited]`

This is a bitmask type.

`_Ios_Fmtflags` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `fmtflags` are:

- `boolalpha`
- `dec`
- `fixed`
- `hex`
- `internal`
- `left`
- `oct`
- `right`
- `scientific`
- `showbase`
- `showpoint`
- `showpos`
- `skipws`
- `unitbuf`
- `uppercase`
- `adjustfield`
- `basefield`
- `floatfield`

Definition at line 323 of file `ios_base.h`.

5.619.2.4 `typedef _Ios_Iostate std::ios_base::iostate` `[inherited]`

This is a bitmask type.

`_Ios_Iostate` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `iostate` are:

- `badbit`
- `eofbit`
- `failbit`
- `goodbit`

Definition at line 398 of file `ios_base.h`.

5.619.2.5 `typedef _Ios_Openmode std::ios_base::openmode` [inherited]

This is a bitmask type.

`_Ios_Openmode` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `openmode` are:

- `app`
- `ate`
- `binary`
- `in`
- `out`
- `trunc`

Definition at line 429 of file `ios_base.h`.

5.619.2.6 `typedef _Ios_Seekdir std::ios_base::seekdir` [inherited]

This is an enumerated type.

`_Ios_Seekdir` is implementation-defined. Defined values of type `seekdir` are:

- `beg`
- `cur`, equivalent to `SEEK_CUR` in the C standard library.
- `end`, equivalent to `SEEK_END` in the C standard library.

Definition at line 461 of file `ios_base.h`.

5.619.3 Member Enumeration Documentation

5.619.3.1 `enum std::ios_base::event` [inherited]

The set of events that may be passed to an event callback.

`erase_event` is used during `~ios()` and `copyfmt()`. `imbue_event` is used during `imbue()`. `copyfmt_event` is used during `copyfmt()`.

Definition at line 487 of file `ios_base.h`.

5.619.4 Constructor & Destructor Documentation

5.619.4.1 `template<typename _CharT, typename _Traits> std::basic_ostream<_CharT, _Traits>::basic_ostream (__streambuf_type* __sb) [inline], [explicit]`

Base constructor.

This ctor is almost never called by the user directly, rather from derived classes' initialization lists, which pass a pointer to their own stream buffer.

Definition at line 84 of file ostream.

5.619.4.2 `template<typename _CharT, typename _Traits> virtual std::basic_ostream<_CharT, _Traits>::~~basic_ostream () [inline], [virtual]`

Base destructor.

This does very little apart from providing a virtual base dtor.

Definition at line 93 of file ostream.

5.619.5 Member Function Documentation

5.619.5.1 `const locale& std::ios_base::M_getloc () const [inline], [inherited]`

Locale access.

Returns

A reference to the current locale.

Like getloc above, but returns a reference instead of generating a copy.

Definition at line 774 of file ios_base.h.

Referenced by `std::time_get<_CharT, _Inlter>::do_date_order()`, `std::time_get<_CharT, _Inlter>::do_get()`, `std::money_get<_CharT, _Inlter>::do_get()`, `std::num_get<_CharT, _Inlter>::do_get()`, `std::time_get<_CharT, _Inlter>::do_get_date()`, `std::time_get<_CharT, _Inlter>::do_get_monthname()`, `std::time_get<_CharT, _Inlter>::do_get_time()`, `std::time_get<_CharT, _Inlter>::do_get_weekday()`, `std::time_get<_CharT, _Inlter>::do_get_year()`, `std::time_put<_CharT, _Outlter>::do_put()`, `std::num_put<_CharT, _Outlter>::do_put()`, `std::time_get<_CharT, _Inlter>::get()`, and `std::time_put<_CharT, _Outlter>::put()`.

5.619.5.2 `template<typename _CharT, typename _Traits> void std::basic_ostream<_CharT, _Traits>::M_write (const char_type* __s, streamsize __n) [inline]`

Core write functionality, without sentry.

Parameters

<code>_↔ _s</code>	The array to insert.
<code>_↔ _n</code>	Maximum number of characters to insert.

Definition at line 311 of file ostream.

Referenced by `std::basic_ostream< _CharT, _Traits >::write()`.

5.619.5.3 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::bad () const`
`[inline], [inherited]`

Fast error checking.

Returns

True if the badbit is set.

Note that other iostate flags may also be set.

Definition at line 211 of file basic_ios.h.

5.619.5.4 `template<typename _CharT, typename _Traits > void std::basic_ios< _CharT, _Traits >::clear (iostate __state =`
`goodbit) [inherited]`

[Re]sets the error state.

Parameters

<code>__state</code>	The new state flag(s) to set.
----------------------	-------------------------------

See `std::ios_base::iostate` for the possible bit values. Most users will not need to pass an argument.

Definition at line 41 of file basic_ios.tcc.

Referenced by `std::basic_ios< char, _Traits >::exceptions()`, `std::__detail::operator>>()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_ios< char, _Traits >::rdstate()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ios< char, _Traits >::setstate()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

5.619.5.5 `template<typename _CharT, typename _Traits > basic_ios< _CharT, _Traits > & std::basic_ios< _CharT, _Traits`
`>::copyfmt (const basic_ios< _CharT, _Traits > & __rhs) [inherited]`

Copies fields of `__rhs` into this.

Parameters

<code>__rhs</code>	The source values for the copies.
--------------------	-----------------------------------

Returns

Reference to this object.

All fields of `__rhs` are copied into this object except that `rdbuf()` and `rdstate()` remain unchanged. All values in the `pword` and `iword` arrays are copied. Before copying, each callback is invoked with `erase_event`. After copying, each (new) callback is invoked with `copyfmt_event`. The final step is to copy exceptions().

Definition at line 63 of file `basic_ios.tcc`.

References `std::basic_ios< _CharT, _Traits >::exceptions()`, `std::basic_ios< _CharT, _Traits >::fill()`, `std::ios_base::flags()`, `std::ios_base::getloc()`, `std::ios_base::precision()`, `std::basic_ios< _CharT, _Traits >::tie()`, `std::tie()`, and `std::ios_base::width()`.

Referenced by `std::basic_ios< char, _Traits >::rdbuf()`.

5.619.5.6 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::eof () const`
`[inline], [inherited]`

Fast error checking.

Returns

True if the eofbit is set.

Note that other `iostate` flags may also be set.

Definition at line 190 of file `basic_ios.h`.

5.619.5.7 `template<typename _CharT, typename _Traits> iostate std::basic_ios< _CharT, _Traits >::exceptions () const`
`[inline], [inherited]`

Throwing exceptions on errors.

Returns

The current exceptions mask.

This changes nothing in the stream. See the one-argument version of `exceptions(iostate)` for the meaning of the return value.

Definition at line 222 of file `basic_ios.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, and `std::basic_ios< char, _Traits >::setstate()`.

5.619.5.8 `template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits >::exceptions (iostate _except)` `[inline], [inherited]`

Throwing exceptions on errors.

Parameters

<code>__except</code>	The new exceptions mask.
-----------------------	--------------------------

By default, error flags are set silently. You can set an exceptions mask for each stream; if a bit in the mask becomes set in the error flags, then an exception of type `std::ios_base::failure` is thrown.

If the error flag is already set when the exceptions mask is added, the exception is immediately thrown. Try running the following under GCC 3.1 or later:

```
1 #include <iostream>
2 #include <fstream>
3 #include <exception>
4
5 int main()
6 {
7     std::set_terminate (__gnu_cxx::__verbose_terminate_handler);
8
9     std::ifstream f ("/etc/motd");
10
11     std::cerr << "Setting badbit\n";
12     f.setstate (std::ios_base::badbit);
13
14     std::cerr << "Setting exception mask\n";
15     f.exceptions (std::ios_base::badbit);
16 }
```

Definition at line 257 of file `basic_ios.h`.

5.619.5.9 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::fail () const`
`[inline], [inherited]`

Fast error checking.

Returns

True if either the badbit or the failbit is set.

Checking the badbit in `fail()` is historical practice. Note that other iostate flags may also be set.

Definition at line 201 of file `basic_ios.h`.

Referenced by `std::basic_ios< char, _Traits >::operator bool()`, `std::basic_ios< char, _Traits >::operator!()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, and `std::regex_traits< _Ch_type >::value()`.

5.619.5.10 `template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::fill () const`
`[inline], [inherited]`

Retrieves the *empty* character.

Returns

The current fill character.

It defaults to a space (' ') in the current locale.

Definition at line 370 of file `basic_ios.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::basic_ios< char, _Traits >::fill()`, `std::normal_distribution< _RealType >::operator()()`, `std::gamma_distribution< _RealType >::operator()()`, `std::negative_binomial_distribution< _IntType >::operator()()`, and `std::operator>>()`.

5.619.5.11 `template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::fill (char_type __ch) [inline],[inherited]`

Sets a new *empty* character.

Parameters

<code>__ch</code>	The new character.
-------------------	--------------------

Returns

The previous fill character.

The fill character is used to fill out space when P+ characters have been requested (e.g., via `setw`), Q characters are actually used, and Q<P. It defaults to a space (' ') in the current locale.

Definition at line 390 of file `basic_ios.h`.

5.619.5.12 `fmtflags std::ios_base::flags () const [inline],[inherited]`

Access to format flags.

Returns

The format control flags for both input and output.

Definition at line 619 of file `ios_base.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >::discard()`, `std::money_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::operator()()`, `std::discard_block_engine< _RandomNumberEngine, __p, __r >::operator()()`, `std::shuffle_order_engine< _RandomNumberEngine, __k >::operator()()`, `std::normal_distribution< _RealType >::operator()()`, `std::gamma_distribution< _RealType >::operator()()`, `std::binomial_distribution< _IntType >::operator()()`, `std::negative_binomial_distribution< _IntType >::operator()()`, `std::poisson_distribution< _IntType >::operator()()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, `std::__detail::operator>>()`, `std::operator>>()`, `std::linear_congruential_engine< _UIntType, __a, __c, __m >::seed()`, and `std::basic_istream< _CharT, _Traits >::sentry::sentry()`.

5.619.5.13 `fmtflags std::ios_base::flags (fmtflags __fmtfl) [inline],[inherited]`

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

Definition at line 630 of file `ios_base.h`.

```
5.619.5.14 template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits> & std::basic_ostream<
    _CharT, _Traits>::flush ( )
```

Synchronizing the stream buffer.

Returns

`*this`

If `rdbuf()` is a null pointer, changes nothing.

Otherwise, calls `rdbuf() -> pubsync()`, and if that returns -1, sets `badbit`.

Definition at line 211 of file `ostream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, `std::basic_ios< _CharT, _Traits>::setstate()`, and `std::basic_ostream< _CharT, _Traits>::tellp()`.

Referenced by `std::operator<<()`, and `std::basic_ostream< _CharT, _Traits>::write()`.

```
5.619.5.15 locale std::ios_base::getloc ( ) const [inline], [inherited]
```

Locale access.

Returns

A copy of the current locale.

If `imbue(loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale()`, the global C++ locale.

Definition at line 763 of file `ios_base.h`.

Referenced by `std::basic_ios< _CharT, _Traits>::copyfmt()`, `std::money_put< _CharT, _Outlter>::do_put()`, `std::operator>>()`, and `std::ws()`.

```
5.619.5.16 template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits>::good ( ) const
    [inline], [inherited]
```

Fast error checking.

Returns

True if no error flags are set.

A wrapper around `rdstate`.

Definition at line 180 of file `basic_ios.h`.

Referenced by `std::__detail::operator>>()`, `std::basic_ostream< _CharT, _Traits>::sentry::sentry()`, and `std::basic_istream< _CharT, _Traits>::sentry::sentry()`.

```
5.619.5.17 template<typename _CharT, typename _Traits> locale std::basic_ios< _CharT, _Traits>::imbue ( const locale &
    __loc ) [inherited]
```

Moves to a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls `ios_base::imbue(loc)`, and if a stream buffer is associated with this stream, calls that buffer's `pubimbue(loc)`.

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>.

Definition at line 114 of file `basic_ios.tcc`.

References `std::ios_base::imbue()`.

Referenced by `std::basic_ios<char, _Traits>::fill()`.

5.619.5.18 `template<typename _CharT, typename _Traits> void std::basic_ios<_CharT, _Traits>::init(basic_streambuf<_CharT, _Traits> * __sb)` `[protected]`, `[inherited]`

All setup is performed here.

This is called from the public constructor. It is not virtual and cannot be redefined.

Definition at line 126 of file `basic_ios.tcc`.

Referenced by `std::basic_ios<char, _Traits>::basic_ios()`.

5.619.5.19 `long& std::ios_base::iword(int __ix)` `[inline]`, `[inherited]`

Access to integer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to an integer associated with the index.

The `iword` function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 809 of file `ios_base.h`.

5.619.5.20 `template<typename _CharT, typename _Traits> char std::basic_ios< _CharT, _Traits >::narrow (char_type __c, char __dfault) const [inline], [inherited]`

Squeezes characters.

Parameters

<code>__c</code>	The character to narrow.
<code>__dfault</code>	The character to narrow.

Returns

The narrowed character.

Maps a character of `char_type` to a character of `char`, if possible.

Returns the result of

```
1 std::use_facet<ctype<char_type> > (getloc()) .narrow(c, dfault)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>.

Definition at line 430 of file `basic_ios.h`.

5.619.5.21 `template<typename _CharT, typename _Traits> std::basic_ios< _CharT, _Traits >::operator bool () const [inline], [explicit], [inherited]`

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`

Definition at line 117 of file `basic_ios.h`.

5.619.5.22 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::operator! () const [inline], [inherited]`

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`

Definition at line 125 of file `basic_ios.h`.

```
5.619.5.23 template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<( __ostream_type &(*)(__ostream_type &)__pf ) [inline]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

Definition at line 108 of file `ostream`.

Referenced by `std::basic_ostream<_CharT, _Traits>::operator<<()`, `std::operator<<()`, and `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`.

```
5.619.5.24 template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<( __ios_type &(*)(__ios_type &)__pf ) [inline]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

Definition at line 117 of file `ostream`.

```
5.619.5.25 template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<( ios_base &(*)(ios_base &)__pf ) [inline]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

Definition at line 127 of file `ostream`.

```
5.619.5.26 template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<( long __n ) [inline]
```

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 166 of file `ostream`.

5.619.5.27 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(unsigned long __n) [inline]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 170 of file `ostream`.

5.619.5.28 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(bool __n) [inline]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 174 of file `ostream`.

5.619.5.29 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::operator<<(short __n)`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 92 of file `ostream.tcc`.

References `std::ios_base::basefield`, `std::ios_base::flags()`, `std::ios_base::hex`, `std::ios_base::oct`, and `std::basic_ostream<_CharT, _Traits>::operator<<()`.

5.619.5.30 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(unsigned short __n) [inline]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 181 of file `ostream`.

5.619.5.31 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::operator<<(int __n)`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 106 of file `ostream.tcc`.

References `std::ios_base::basefield`, `std::ios_base::flags()`, `std::ios_base::hex`, `std::ios_base::oct`, and `std::basic_ostream<_CharT, _Traits>::operator<<()`.

5.619.5.32 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(unsigned int __n) [inline]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 192 of file `ostream`.

5.619.5.33 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(long long __n) [inline]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 201 of file `ostream`.

5.619.5.34 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(unsigned long long __n) [inline]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 205 of file ostream.

```
5.619.5.35 template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<( double __f ) [inline]
```

Floating point arithmetic inserters.

Parameters

↵	A variable of builtin floating point type.
↵	
↵	
↵	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 220 of file ostream.

```
5.619.5.36 template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<( float __f ) [inline]
```

Floating point arithmetic inserters.

Parameters

↵	A variable of builtin floating point type.
↵	
↵	
↵	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 224 of file ostream.

5.619.5.37 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<< (long double __f) [inline]`

Floating point arithmetic inserters.

Parameters

<code>↔</code>	A variable of builtin floating point type.
<code>__↔</code>	
<code>↔</code>	
<code>__↔</code>	
<code>f</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 232 of file `ostream`.

5.619.5.38 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<< (const void * __p) [inline]`

Pointer arithmetic inserters.

Parameters

<code>__↔</code>	A variable of pointer type.
<code>__p</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 245 of file `ostream`.

5.619.5.39 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::operator<< (__streambuf_type * __sb)`

Extracting from another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is `NULL`, the stream will set `failbit` in its error state.

Characters are extracted from `__sb` and inserted into `*this` until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output sequence fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs while getting a character from `__sb`, which sets `failbit` in the error state

If the function inserts no characters, `failbit` is set.

Definition at line 120 of file `ostream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

5.619.5.40 `streamsize std::ios_base::precision () const` `[inline],[inherited]`

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Definition at line 689 of file `ios_base.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::normal_distribution< _RealType >::operator()()`, `std::gamma_distribution< _RealType >::operator()()`, `std::negative_binomial_distribution< _IntType >::operator()()`, and `std::operator>>()`.

5.619.5.41 `streamsize std::ios_base::precision (streamsize __prec)` `[inline],[inherited]`

Changing flags.

Parameters

<code>__prec</code>	The new precision value.
---------------------	--------------------------

Returns

The previous value of `precision()`.

Definition at line 698 of file `ios_base.h`.

5.619.5.42 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::put (char_type __c)`

Simple insertion.

Parameters

<code>__c</code>	The character to insert.
------------------	--------------------------

Returns

`*this`

Tries to insert `__c`.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 149 of file ostream.tcc.

References `std::ios_base::badbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, `std::basic_ios< _CharT, _Traits>::setstate()`, and `std::basic_ostream< _CharT, _Traits>::write()`.

Referenced by `std::basic_ostream< _CharT, _Traits>::operator<<()`.

5.619.5.43 `void*& std::ios_base::pword (int __ix) [inline], [inherited]`

Access to void pointer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to a `void*` associated with the index.

The `pword` function provides access to an array of pointers that can be used for any purpose. The array grows as required to hold the supplied index. All pointers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 830 of file ios_base.h.

5.619.5.44 `template<typename _CharT, typename _Traits> basic_streambuf< _CharT, _Traits>* std::basic_ios< _CharT, _Traits>::rdbuf () const` `[inline], [inherited]`

Accessing the underlying buffer.

Returns

The current stream buffer.

This does not change the state of the stream.

Definition at line 321 of file `basic_ios.h`.

Referenced by `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::tr2::operator>>()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_ios< char, _Traits >::rdbuf()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_istream< _CharT, _Traits >::sentry()`, `std::sentry()`, `std::basic_istream< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::ws()`.

5.619.5.45 `template<typename _CharT, typename _Traits> basic_streambuf< _CharT, _Traits> * std::basic_ios< _CharT, _Traits>::rdbuf (basic_streambuf< _CharT, _Traits> * __sb)` `[inherited]`

Changing the underlying buffer.

Parameters

<code>__sb</code>	The new stream buffer.
-------------------	------------------------

Returns

The previous stream buffer.

Associates a new buffer with the current stream, and clears the error state.

Due to historical accidents which the LWG refuses to correct, the I/O library suffers from a design error: this function is hidden in derived classes by overrides of the zero-argument `rdbuf()`, which is non-virtual for hysterical raisins. As a result, you must use explicit qualifications to access this function via any derived class. For example:

```
1 std::fstream    foo;           // or some other derived type
2 std::streambuf* p = .....;
3
4 foo.ios::rdbuf(p);           // ios == basic_ios<char>
```

Definition at line 53 of file `basic_ios.tcc`.

5.619.5.46 `template<typename _CharT, typename _Traits> iosstate std::basic_ios< _CharT, _Traits >::rdstate () const`
`[inline], [inherited]`

Returns the error state of the stream buffer.

Returns

A bit pattern (well, isn't everything?)

See `std::ios_base::iosstate` for the possible bit values. Most users will call one of the interpreting wrappers, e.g., `good()`.

Definition at line 137 of file `basic_ios.h`.

Referenced by `std::basic_ios< char, _Traits >::bad()`, `std::basic_ios< char, _Traits >::eof()`, `std::basic_ios< char, _Traits >::fail()`, `std::basic_ios< char, _Traits >::good()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ios< char, _Traits >::setstate()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

5.619.5.47 `void std::ios_base::register_callback (event_callback __fn, int __index)` `[inherited]`

Add the callback `__fn` with parameter `__index`.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

5.619.5.48 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::seekp (pos_type __pos)`

Changing the current write position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

`*this`

If `fail()` is not true, calls `rddbuf() -> pubseekpos(pos)`. If that function fails, sets failbit.

Definition at line 258 of file `ostream.tcc`.

References `std::ios_base::badbit`, `std::basic_ios< _CharT, _Traits>::fail()`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::ios_base::out`, `std::basic_ios< _CharT, _Traits>::rddbuf()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

Referenced by `std::basic_ostream< _CharT, _Traits>::tellp()`.

5.619.5.49 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::seekp (off_type __off, ios_base::seekdir __dir)`

Changing the current write position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

`*this`

If `fail()` is not true, calls `rdbuf()->pubseekoff(off, dir)`. If that function fails, sets failbit.

Definition at line 290 of file `ostream.tcc`.

References `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::ios_base::out`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

5.619.5.50 `fmtflags std::ios_base::setf (fmtflags __fmtfl) [inline], [inherited]`

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
----------------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set.

Definition at line 646 of file `ios_base.h`.

Referenced by `std::boolalpha()`, `std::dec()`, `std::fixed()`, `std::hex()`, `std::hexfloat()`, `std::left()`, `std::oct()`, `std::__detail<::operator>>()`, `std::right()`, `std::scientific()`, `std::showbase()`, `std::showpoint()`, `std::showpos()`, `std::skipws()`, `std::unitbuf()`, and `std::uppercase()`.

5.619.5.51 `fmtflags std::ios_base::setf (fmtflags __fmtfl, fmtflags __mask) [inline], [inherited]`

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
<code>__mask</code>	The flags mask for <code>__fmtfl</code> .

Returns

The previous format control flags.

This function clears *mask* in the format flags, then sets *fmtfl* & *mask*. An example mask is `ios_base::adjustfield`.

Definition at line 663 of file `ios_base.h`.

5.619.5.52 `template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits >::setstate (iostate __state)`
`[inline], [inherited]`

Sets additional flags in the error state.

Parameters

<code>__state</code>	The additional state flag(s) to set.
----------------------	--------------------------------------

See `std::ios_base::iostate` for the possible bit values.

Definition at line 157 of file `basic_ios.h`.

Referenced by `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::tr2::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::ws()`.

5.619.5.53 `static bool std::ios_base::sync_with_stdio (bool __sync = true)` `[static], [inherited]`

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., `stdout`) and the standard C++ objects (e.g., `cout`). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstream.html#std.io.filestreams.binary>

5.619.554 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>::pos_type
std::basic_ostream<_CharT, _Traits>::tellp ()`

Getting the current write position.

Returns

A file position object.

If `fail()` is not false, returns `pos_type(-1)` to indicate failure. Otherwise returns `rdbuf() -> pubseekoff(0, cur, out)`.

Definition at line 237 of file `ostream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::cur`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::ios_base::out`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ostream<_CharT, _Traits>::seekp()`.

Referenced by `std::basic_ostream<_CharT, _Traits>::flush()`.

5.619.555 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>* std::basic_ios<_CharT,
_Traits>::tie () const [inline], [inherited]`

Fetches the current *tied* stream.

Returns

A pointer to the tied stream, or NULL if the stream is not tied.

A stream may be *tied* (or synchronized) to a second output stream. When this stream performs any I/O, the tied stream is first flushed. For example, `std::cin` is tied to `std::cout`.

Definition at line 295 of file `basic_ios.h`.

Referenced by `std::basic_ios<char, _Traits>::basic_ios()`, `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`, and `std::basic_istream<_CharT, _Traits>::sentry::sentry()`.

5.619.556 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>* std::basic_ios<_CharT,
_Traits>::tie (basic_ostream<_CharT, _Traits>* __tiestr) [inline], [inherited]`

Ties this stream to an output stream.

Parameters

<code>__tiestr</code>	The output stream.
-----------------------	--------------------

Returns

The previously tied output stream, or NULL if the stream was not tied.

This sets up a new tie; see `tie()` for more.

Definition at line 307 of file `basic_ios.h`.

5.619.5.57 `void std::ios_base::unsetf (fmtflags __mask)` `[inline],[inherited]`

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

Definition at line 678 of file `ios_base.h`.

Referenced by `std::defaultfloat()`, `std::noboolalpha()`, `std::noshowbase()`, `std::noshowpoint()`, `std::noshowpos()`, `std::noskipws()`, `std::nounitbuf()`, and `std::nouppercase()`.

5.619.5.58 `template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::widen (char __c)`
`const` `[inline],[inherited]`

Widens characters.

Parameters

<code>__c</code>	The character to widen.
------------------	-------------------------

Returns

The widened character.

Maps a character of `char` to a character of `char_type`.

Returns the result of

```
1 std::use_facet<ctype<char_type>> >(getloc()).widen(c)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Definition at line 449 of file `basic_ios.h`.

Referenced by `std::basic_ios< char, _Traits >::fill()`, `std::getline()`, `std::normal_distribution< _RealType >::operator()()`, `std::gamma_distribution< _RealType >::operator()()`, `std::negative_binomial_distribution< _IntType >::operator()()`, `std::tr2::operator>>()`, and `std::operator>>()`.

5.619.5.59 **streamsize** std::ios_base::width() const [inline],[inherited]

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Definition at line 712 of file ios_base.h.

Referenced by std::basic_ios< _CharT, _Traits >::copyfmt(), std::money_get< _CharT, _Inlter >::do_get(), std::num_get< _CharT, _Inlter >::do_get(), std::num_put< _CharT, _Outlter >::do_put(), and std::operator>>().

5.619.5.60 **streamsize** std::ios_base::width(streamsize __wide) [inline],[inherited]

Changing flags.

Parameters

__wide	The new width value.
--------	----------------------

Returns

The previous value of width().

Definition at line 721 of file ios_base.h.

5.619.5.61 **template**<typename _CharT, typename _Traits> **basic_ostream**< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::write(const char_type * __s, streamsize __n)

Character string insertion.

Parameters

\leftarrow __s	The array to insert.
\leftarrow __n	Maximum number of characters to insert.

Returns

*this

Characters are copied from __s and inserted into the stream until one of the following happens:

- `__n` characters are inserted
- inserting into the output sequence fails (in this case, `badbit` will be set in the stream's error state)

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 183 of file `ostream.tcc`.

References `std::basic_ostream< _CharT, _Traits >::_M_write()`, `std::ios_base::badbit`, and `std::basic_ostream< _CharT, _Traits >::flush()`.

Referenced by `std::basic_ostream< _CharT, _Traits >::put()`.

5.619.5.62 `static int std::ios_base::xalloc () throw` `[static], [inherited]`

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the `iword` and `pword` functions. The expectation is that an application calls `xalloc` in order to obtain an index in the `iword` and `pword` arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. `xalloc` is guaranteed to return an index that is safe to use in the `iword` and `pword` arrays.

5.619.6 Member Data Documentation

5.619.6.1 `const fmtflags std::ios_base::adjustfield` `[static], [inherited]`

A mask of `left|right|internal`. Useful for the 2-arg form of `setf`.

Definition at line 378 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _Inlter >::do_get()`, `std::num_put< _CharT, _Outlter >::do_put()`, `std::internal()`, `std::left()`, and `std::right()`.

5.619.6.2 `const openmode std::ios_base::app` `[static], [inherited]`

Seek to end before each write.

Definition at line 432 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

5.619.6.3 const openmode std::ios_base::ate [static],[inherited]

Open and seek to end immediately after opening.

Definition at line 435 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::open().

5.619.6.4 const iostate std::ios_base::badbit [static],[inherited]

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Definition at line 402 of file ios_base.h.

Referenced by std::basic_ios< char, _Traits >::bad(), std::basic_ios< char, _Traits >::fail(), std::basic_ostream< \leftarrow _CharT, _Traits >::flush(), std::basic_istream< _CharT, _Traits >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::basic_istream< _CharT, _Traits >::ignore(), std::basic_ostream< _CharT, _Traits >::operator<<(), std::operator<<(), std::basic_istream< _CharT, _Traits >::operator>>(), std::operator>>(), std::basic_istream< \leftarrow _CharT, _Traits >::peek(), std::basic_ostream< _CharT, _Traits >::put(), std::basic_istream< _CharT, _Traits >::putback(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::readsome(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_ostream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< \leftarrow CharT, _Traits >::sync(), std::basic_istream< _CharT, _Traits >::tellg(), std::basic_ostream< _CharT, _Traits >::tellp(), std::basic_istream< _CharT, _Traits >::unget(), and std::basic_ostream< _CharT, _Traits >::write().

5.619.6.5 const fmtflags std::ios_base::basefield [static],[inherited]

A mask of dec|oct|hex. Useful for the 2-arg form of setf.

Definition at line 381 of file ios_base.h.

Referenced by std::dec(), std::num_get< _CharT, _InIter >::do_get(), std::num_put< _CharT, _OutIter >::do_put(), std::hex(), std::oct(), and std::basic_ostream< _CharT, _Traits >::operator<<().

5.619.6.6 const seekdir std::ios_base::beg [static],[inherited]

Request a seek relative to the beginning of the stream.

Definition at line 464 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::seekpos(), and __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync().

5.619.6.7 const openmode std::ios_base::binary [static],[inherited]

Perform input and output in binary mode (as opposed to text mode). This is probably not what you think it is; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.←filestreams.binary>.

Definition at line 440 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::showmanyc().

5.619.6.8 `const fmtflags std::ios_base::boolalpha` `[static], [inherited]`

Insert/extract `bool` in alphabetic rather than numeric format.

Definition at line 326 of file `ios_base.h`.

Referenced by `std::boolalpha()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_put<_CharT, _OutIter>::do_put()`, and `std::noboolalpha()`.

5.619.6.9 `const seekdir std::ios_base::cur` `[static], [inherited]`

Request a seek relative to the current position within the sequence.

Definition at line 467 of file `ios_base.h`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::imbue()`, `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_filebuf<_CharT, _Traits>::pbackfail()`, `std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekoff()`, `std::basic_filebuf<_CharT, _Traits>::seekoff()`, `__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>::sync()`, `std::basic_istream<_CharT, _Traits>::tellg()`, and `std::basic_ostream<_CharT, _Traits>::tellp()`.

5.619.6.10 `const fmtflags std::ios_base::dec` `[static], [inherited]`

Converts integer input or generates integer output in decimal base.

Definition at line 329 of file `ios_base.h`.

Referenced by `std::dec()`.

5.619.6.11 `const seekdir std::ios_base::end` `[static], [inherited]`

Request a seek relative to the current end of the sequence.

Definition at line 470 of file `ios_base.h`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::open()`, and `std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekoff()`.

5.619.6.12 `const iostate std::ios_base::eofbit` `[static], [inherited]`

Indicates that an input operation reached the end of an input sequence.

Definition at line 405 of file `ios_base.h`.

Referenced by `std::time_get<_CharT, _InIter>::do_get()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::time_get<_CharT, _InIter>::do_get_date()`, `std::time_get<_CharT, _InIter>::do_get_monthname()`, `std::time_get<_CharT, _InIter>::do_get_time()`, `std::time_get<_CharT, _InIter>::do_get_weekday()`, `std::time_get<_CharT, _InIter>::do_get_year()`, `std::basic_ios<char, _Traits>::eof()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::time_get<_CharT, _InIter>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::operator>>()`, `std::basic_istream<_CharT, _Traits>::peek()`, `std::basic_istream<_CharT, _Traits>::putback()`, `std::basic_istream<_CharT, _Traits>::read()`, `std::basic_istream<_CharT, _Traits>::readsome()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_istream<_CharT, _Traits>::unget()`, and `std::ws()`.

5.619.6.13 const iostate std::ios_base::failbit [static],[inherited]

Indicates that an input operation failed to read the expected characters, or that an output operation failed to generate the desired characters.

Definition at line 410 of file ios_base.h.

Referenced by std::time_get< _CharT, _Inlter >::do_date_order(), std::num_get< _CharT, _Inlter >::do_get(), std::time_get< _CharT, _Inlter >::do_get_monthname(), std::time_get< _CharT, _Inlter >::do_get_weekday(), std::time_get< _CharT, _Inlter >::do_get_year(), std::basic_ios< char, _Traits >::fail(), std::basic_istream< _CharT, _Traits >::get(), std::time_get< _CharT, _Inlter >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::basic_ostream< _CharT, _Traits >::operator<<(), std::basic_istream< _CharT, _Traits >::operator>>(), std::operator>>(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_ostream< _CharT, _Traits >::sentry::sentry(), and std::basic_istream< _CharT, _Traits >::sentry::sentry().

5.619.6.14 const fmtflags std::ios_base::fixed [static],[inherited]

Generate floating-point output in fixed-point notation.

Definition at line 332 of file ios_base.h.

Referenced by std::num_get< _CharT, _Inlter >::do_get(), std::fixed(), and std::hexfloat().

5.619.6.15 const fmtflags std::ios_base::floatfield [static],[inherited]

A mask of scientific|fixed. Useful for the 2-arg form of setf.

Definition at line 384 of file ios_base.h.

Referenced by std::defaultfloat(), std::num_get< _CharT, _Inlter >::do_get(), std::fixed(), std::hexfloat(), and std::scientific().

5.619.6.16 const iostate std::ios_base::goodbit [static],[inherited]

Indicates all is well.

Definition at line 413 of file ios_base.h.

Referenced by std::time_get< _CharT, _Inlter >::do_date_order(), std::time_get< _CharT, _Inlter >::do_get(), std::num_get< _CharT, _Inlter >::do_get(), std::time_get< _CharT, _Inlter >::do_get_monthname(), std::time_get< _CharT, _Inlter >::do_get_weekday(), std::time_get< _CharT, _Inlter >::do_get_year(), std::basic_ostream< _CharT, _Traits >::flush(), std::basic_istream< _CharT, _Traits >::get(), std::time_get< _CharT, _Inlter >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::basic_istream< _CharT, _Traits >::ignore(), std::basic_ostream< _CharT, _Traits >::operator<<(), std::basic_istream< _CharT, _Traits >::operator>>(), std::operator>>(), std::basic_istream< _CharT, _Traits >::peek(), std::basic_ostream< _CharT, _Traits >::put(), std::basic_istream< _CharT, _Traits >::putback(), std::basic_ios< char, _Traits >::rdstate(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::readsome(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_ostream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::sync(), and std::basic_istream< _CharT, _Traits >::unset().

5.619.6.17 const fmtflags std::ios_base::hex [static],[inherited]

Converts integer input or generates integer output in hexadecimal base.

Definition at line 335 of file ios_base.h.

Referenced by std::num_get< _CharT, _Inlter >::do_get(), std::num_put< _CharT, _Outlter >::do_put(), std::hex(), and std::basic_ostream< _CharT, _Traits >::operator<<().

5.619.6.18 const openmode std::ios_base::in [static],[inherited]

Open for input. Default for ifstream and fstream.

Definition at line 443 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::pbackfail(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos(), std::basic_filebuf< _CharT, _Traits >::showmanyc(), __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync(), std::basic_istream< _CharT, _Traits >::tellg(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::underflow(), std::basic_filebuf< _CharT, _Traits >::underflow(), and std::basic_filebuf< _CharT, _Traits >::xsgetn().

5.619.6.19 const fmtflags std::ios_base::internal [static],[inherited]

Adds fill characters at a designated internal point in certain generated output, or identical to right if no such point is designated.

Definition at line 340 of file ios_base.h.

Referenced by std::money_get< _CharT, _Inlter >::do_get(), std::num_put< _CharT, _Outlter >::do_put(), and std::internal().

5.619.6.20 const fmtflags std::ios_base::left [static],[inherited]

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.)

Definition at line 344 of file ios_base.h.

Referenced by std::money_get< _CharT, _Inlter >::do_get(), std::num_put< _CharT, _Outlter >::do_put(), and std::left().

5.619.6.21 const fmtflags std::ios_base::oct [static],[inherited]

Converts integer input or generates integer output in octal base.

Definition at line 347 of file ios_base.h.

Referenced by std::num_get< _CharT, _Inlter >::do_get(), std::oct(), and std::basic_ostream< _CharT, _Traits >::operator<<().

5.619.6.22 const openmode std::ios_base::out [static],[inherited]

Open for output. Default for `ofstream` and `fstream`.

Definition at line 446 of file `ios_base.h`.

Referenced by `std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow()`, `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::pbackfail()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos()`, `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

5.619.6.23 const fmtflags std::ios_base::right [static],[inherited]

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Definition at line 351 of file `ios_base.h`.

Referenced by `std::right()`.

5.619.6.24 const fmtflags std::ios_base::scientific [static],[inherited]

Generates floating-point output in scientific notation.

Definition at line 354 of file `ios_base.h`.

Referenced by `std::hexfloat()`, and `std::scientific()`.

5.619.6.25 const fmtflags std::ios_base::showbase [static],[inherited]

Generates a prefix indicating the numeric base of generated integer output.

Definition at line 358 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::noshowbase()`, and `std::showbase()`.

5.619.6.26 const fmtflags std::ios_base::showpoint [static],[inherited]

Generates a decimal-point character unconditionally in generated floating-point output.

Definition at line 362 of file `ios_base.h`.

Referenced by `std::noshowpoint()`, and `std::showpoint()`.

5.619.6.27 const fmtflags std::ios_base::showpos [static],[inherited]

Generates a + sign in non-negative generated numeric output.

Definition at line 365 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _InIter >::do_get()`, `std::noshowpos()`, and `std::showpos()`.

5.619.6.28 `const fmtflags std::ios_base::skipws` `[static], [inherited]`

Skips leading white space before certain input operations.

Definition at line 368 of file `ios_base.h`.

Referenced by `std::noskipws()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, and `std::skipws()`.

5.619.6.29 `const openmode std::ios_base::trunc` `[static], [inherited]`

Open for input. Default for `ofstream`.

Definition at line 449 of file `ios_base.h`.

5.619.6.30 `const fmtflags std::ios_base::unitbuf` `[static], [inherited]`

Flushes output after each output operation.

Definition at line 371 of file `ios_base.h`.

Referenced by `std::nounitbuf()`, and `std::unitbuf()`.

5.619.6.31 `const fmtflags std::ios_base::uppercase` `[static], [inherited]`

Replaces certain lowercase letters with their uppercase equivalents in generated output.

Definition at line 375 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::noskipws()`, and `std::uppercase()`.

The documentation for this class was generated from the following files:

- [iosfwd](#)
- [ostream](#)
- [ostream.tcc](#)

5.620 `std::basic_ostream< _CharT, _Traits >::sentry` Class Reference

Public Member Functions

- [sentry](#) ([basic_ostream](#)< `_CharT`, `_Traits` > &__os)
- [~sentry](#) ()
- [operator bool](#) () const

5.620.1 Detailed Description

```
template<typename _CharT, typename _Traits>
class std::basic_ostream<_CharT, _Traits>::sentry
```

Performs setup work for output streams.

Objects of this class are created before all of the standard inserters are run. It is responsible for *exception-safe prefix and suffix operations*.

Definition at line 426 of file ostream.

5.620.2 Constructor & Destructor Documentation

```
5.620.2.1 template<typename _CharT, typename _Traits> std::basic_ostream<_CharT, _Traits>::sentry (
    basic_ostream<_CharT, _Traits> &__os ) [explicit]
```

The constructor performs preparatory work.

Parameters

<code>__os</code>	The output stream to guard.
-------------------	-----------------------------

If the stream state is good (`__os.good()` is true), then if the stream is tied to another output stream, `is->tie()->flush()` is called to synchronize the output sequences.

If the stream state is still good, then the sentry state becomes true (*okay*).

Definition at line 47 of file ostream.tcc.

References `std::ios_base::badbit`, `std::ios_base::failbit`, `std::basic_ios<_CharT, _Traits>::good()`, `std::ios_base::goodbit`, `std::basic_ostream<_CharT, _Traits>::operator<<()`, `std::num_put<_CharT, _Outlter>::put()`, `std::basic_ios<_CharT, _Traits>::setstate()`, and `std::basic_ios<_CharT, _Traits>::tie()`.

```
5.620.2.2 template<typename _CharT, typename _Traits> std::basic_ostream<_CharT, _Traits>::~sentry::~sentry ( )
    [inline]
```

Possibly flushes the stream.

If `ios_base::unitbuf` is set in `os.flags()`, and `std::uncaught_exception()` is true, the sentry destructor calls `flush()` on the output stream.

Definition at line 454 of file ostream.

5.620.3 Member Function Documentation

5.620.3.1 `template<typename _CharT, typename _Traits> std::basic_ostream< _CharT, _Traits >::sentry::operator bool ()`
`const [inline], [explicit]`

Quick status checking.

Returns

The sentry state.

For ease of use, sentries may be converted to booleans. The return value is that of the sentry state (true == okay).

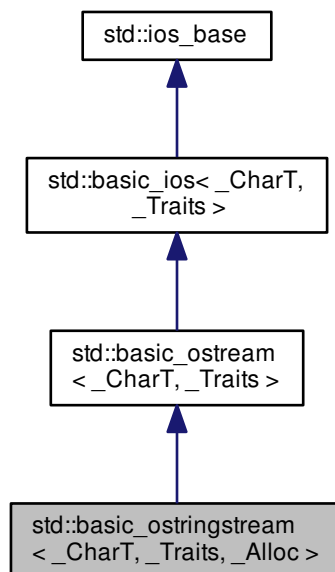
Definition at line 475 of file ostream.

The documentation for this class was generated from the following files:

- [ostream](#)
- [ostream.tcc](#)

5.621 `std::basic_ostringstream< _CharT, _Traits, _Alloc >` Class Template Reference

Inheritance diagram for `std::basic_ostringstream< _CharT, _Traits, _Alloc >`:



Public Types

- typedef [ctype](#)< _CharT > **__ctype_type**
 - typedef [basic_ios](#)< _CharT, _Traits > **__ios_type**
 - typedef [num_put](#)< _CharT, [ostreambuf_iterator](#)< _CharT, _Traits > > **__num_put_type**
 - typedef [basic_ostream](#)< char_type, traits_type > **__ostream_type**
 - typedef [basic_streambuf](#)< _CharT, _Traits > **__streambuf_type**
 - typedef [basic_string](#)< _CharT, _Traits, _Alloc > **__string_type**
 - typedef [basic_stringbuf](#)< _CharT, _Traits, _Alloc > **__stringbuf_type**
 - typedef _Alloc **allocator_type**
 - typedef _CharT **char_type**
 - enum [event](#) { [erase_event](#), [imbue_event](#), [copyfmt_event](#) }
 - typedef void(* [event_callback](#)) (event __e, [ios_base](#) &__b, int __i)
 - typedef _Ios_Fmtflags **fmtflags**
 - typedef traits_type::int_type **int_type**
 - typedef int **io_state**
 - typedef _Ios_istate **istate**
 - typedef traits_type::off_type **off_type**
 - typedef int **open_mode**
 - typedef _Ios_Openmode **openmode**
 - typedef traits_type::pos_type **pos_type**
 - typedef int **seek_dir**
 - typedef _Ios_Seekdir **seekdir**
 - typedef [std::streamoff](#) **streamoff**
 - typedef [std::streampos](#) **streampos**
 - typedef _Traits **traits_type**
-
- typedef [num_get](#)< _CharT, [istreambuf_iterator](#)< _CharT, _Traits > > **__num_get_type**

Public Member Functions

- [basic_ostringstream](#) ([ios_base::openmode](#) __mode=[ios_base::out](#))
- [basic_ostringstream](#) (const [__string_type](#) &__str, [ios_base::openmode](#) __mode=[ios_base::out](#))
- **basic_ostringstream** (const [basic_ostringstream](#) &)=delete
- **basic_ostringstream** ([basic_ostringstream](#) &&__rhs)
- [~basic_ostringstream](#) ()
- const [locale](#) & [_M_getloc](#) () const
- template<typename _ValueT >
 [basic_ostream](#)< _CharT, _Traits > & [_M_insert](#) (_ValueT __v)
- void [_M_setstate](#) ([istate](#) __state)
- bool [bad](#) () const
- void [clear](#) ([istate](#) __state=[goodbit](#))
- [basic_ios](#) & [copyfmt](#) (const [basic_ios](#) &__rhs)
- bool [eof](#) () const
- [istate exceptions](#) () const
- void [exceptions](#) ([istate](#) __except)
- bool [fail](#) () const
- char_type [fill](#) () const
- char_type [fill](#) (char_type __ch)

- `fmtflags flags () const`
 - `fmtflags flags (fmtflags __fmtfl)`
 - `__ostream_type & flush ()`
 - `locale getloc () const`
 - `bool good () const`
 - `locale imbue (const locale & __loc)`
 - `long & iword (int __ix)`
 - `char narrow (char_type __c, char __default) const`
 - `__ostream_type & operator<< (const void * __p)`
 - `__ostream_type & operator<< (__streambuf_type * __sb)`
 - `basic_ostringstream & operator= (const basic_ostringstream &)=delete`
 - `basic_ostringstream & operator= (basic_ostringstream && __rhs)`
 - `streamsize precision () const`
 - `streamsize precision (streamsize __prec)`
 - `void *& pword (int __ix)`
 - `basic_streambuf< _CharT, _Traits > * rdbuf (basic_streambuf< _CharT, _Traits > * __sb)`
 - `__stringbuf_type * rdbuf () const`
 - `iostate rdstate () const`
 - `void register_callback (event_callback __fn, int __index)`
 - `__ostream_type & seekp (pos_type)`
 - `__ostream_type & seekp (off_type, ios_base::seekdir)`
 - `fmtflags setf (fmtflags __fmtfl)`
 - `fmtflags setf (fmtflags __fmtfl, fmtflags __mask)`
 - `void setstate (iostate __state)`
 - `__string_type str () const`
 - `void str (const __string_type & __s)`
 - `void swap (basic_ostringstream & __rhs)`
 - `pos_type tellp ()`
 - `basic_ostream< _CharT, _Traits > * tie () const`
 - `basic_ostream< _CharT, _Traits > * tie (basic_ostream< _CharT, _Traits > * __tiestr)`
 - `void unsetf (fmtflags __mask)`
 - `char_type widen (char __c) const`
 - `streamsize width () const`
 - `streamsize width (streamsize __wide)`
-
- `__ostream_type & operator<< (__ostream_type &(*__pf)(__ostream_type &))`
 - `__ostream_type & operator<< (__ios_type &(*__pf)(__ios_type &))`
 - `__ostream_type & operator<< (ios_base &(*__pf)(ios_base &))`

Inserters

All the `operator<<` functions (aka formatted output functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This can have several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- `__ostream_type & operator<< (long __n)`
- `__ostream_type & operator<< (unsigned long __n)`
- `__ostream_type & operator<< (bool __n)`

- `__ostream_type` & `operator<<` (short __n)
- `__ostream_type` & `operator<<` (unsigned short __n)
- `__ostream_type` & `operator<<` (int __n)
- `__ostream_type` & `operator<<` (unsigned int __n)
- `__ostream_type` & `operator<<` (long long __n)
- `__ostream_type` & `operator<<` (unsigned long long __n)
- `__ostream_type` & `operator<<` (double __f)
- `__ostream_type` & `operator<<` (float __f)
- `__ostream_type` & `operator<<` (long double __f)

Unformatted Output Functions

All the unformatted output functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This has several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state. If `badbit` is on in the stream's exceptions mask, the exception will be rethrown without completing its actions.

- `__ostream_type` & `put` (char_type __c)
- void `_M_write` (const char_type *__s, streamsize __n)
- `__ostream_type` & `write` (const char_type *__s, streamsize __n)
- `operator bool` () const
- bool `operator!` () const

Static Public Member Functions

- static bool `sync_with_stdio` (bool __sync=true)
- static int `xalloc` () throw ()

Static Public Attributes

- static const `fmtflags` `adjustfield`
- static const `openmode` `app`
- static const `openmode` `ate`
- static const `iosstate` `badbit`
- static const `fmtflags` `basefield`
- static const `seekdir` `beg`
- static const `openmode` `binary`
- static const `fmtflags` `boolalpha`
- static const `seekdir` `cur`
- static const `fmtflags` `dec`
- static const `seekdir` `end`
- static const `iosstate` `eofbit`
- static const `iosstate` `failbit`
- static const `fmtflags` `fixed`
- static const `fmtflags` `floatfield`

- static const [iostate](#) [goodbit](#)
- static const [fmtflags](#) [hex](#)
- static const [openmode](#) [in](#)
- static const [fmtflags](#) [internal](#)
- static const [fmtflags](#) [left](#)
- static const [fmtflags](#) [oct](#)
- static const [openmode](#) [out](#)
- static const [fmtflags](#) [right](#)
- static const [fmtflags](#) [scientific](#)
- static const [fmtflags](#) [showbase](#)
- static const [fmtflags](#) [showpoint](#)
- static const [fmtflags](#) [showpos](#)
- static const [fmtflags](#) [skipws](#)
- static const [openmode](#) [trunc](#)
- static const [fmtflags](#) [unitbuf](#)
- static const [fmtflags](#) [uppercase](#)

Protected Types

- enum { [_S_local_word_size](#) }

Protected Member Functions

- void [_M_cache_locale](#) (const [locale](#) & __loc)
- void [_M_call_callbacks](#) ([event](#) __ev) throw ()
- void [_M_dispose_callbacks](#) (void) throw ()
- [_Words](#) & [_M_grow_words](#) (int __index, bool __iword)
- void [_M_init](#) () throw ()
- template<typename [_ValueT](#) >
[__ostream_type](#) & [_M_insert](#) ([_ValueT](#) __v)
- void [_M_move](#) ([ios_base](#) &) noexcept
- void [_M_swap](#) ([ios_base](#) & __rhs) noexcept
- void [init](#) ([basic_streambuf](#)< [_CharT](#), [_Traits](#) > *__sb)
- void [move](#) ([basic_ios](#) & __rhs)
- void [move](#) ([basic_ios](#) && __rhs)
- void [set_rdbuf](#) ([basic_streambuf](#)< [_CharT](#), [_Traits](#) > *__sb)
- void [swap](#) ([basic_ostream](#) & __rhs)
- void [swap](#) ([basic_ios](#) & __rhs) noexcept

Protected Attributes

- [_Callback_list](#) * [_M_callbacks](#)
- const [__ctype_type](#) * [_M_ctype](#)
- [iostate](#) [_M_exception](#)
- [char_type](#) [_M_fill](#)
- bool [_M_fill_init](#)
- [fmtflags](#) [_M_flags](#)
- [locale](#) [_M_ios_locale](#)

- `_Words` `_M_local_word` [`_S_local_word_size`]
- `const` `__num_get_type` * `_M_num_get`
- `const` `__num_put_type` * `_M_num_put`
- `streamsize` `_M_precision`
- `basic_streambuf`< `_CharT`, `_Traits` > * `_M_streambuf`
- `iosstate` `_M_streambuf_state`
- `basic_ostream`< `_CharT`, `_Traits` > * `_M_tie`
- `streamsize` `_M_width`
- `_Words` * `_M_word`
- `int` `_M_word_size`
- `_Words` `_M_word_zero`

5.621.1 Detailed Description

```
template<typename _CharT, typename _Traits, typename _Alloc>
class std::basic_ostringstream< _CharT, _Traits, _Alloc >
```

Controlling output for `std::string`.

Template Parameters

<code>_CharT</code>	Type of character stream.
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_CharT></code> .

This class supports writing to objects of type `std::basic_string`, using the inherited functions from `std::basic_ostream`. To control the associated sequence, an instance of `std::basic_stringbuf` is used, which this page refers to as `sb`.

Definition at line 104 of file `iosfwd`.

5.621.2 Member Typedef Documentation

5.621.2.1 `template<typename _CharT, typename _Traits> typedef num_get<_CharT, istreambuf_iterator<_CharT, _Traits>> __std::basic_ios<_CharT, _Traits>::__num_get_type` [inherited]

These are non-standard types.

Definition at line 91 of file `basic_ios.h`.

5.621.2.2 `typedef void(* std::ios_base::event_callback)(event __e, ios_base &__b, int __i)` [inherited]

The type of an event callback function.

Parameters

<code>_↔ _e</code>	One of the members of the event enum.
<code>_↔ _b</code>	Reference to the ios_base object.
<code>_↔ _i</code>	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several ios_base and basic_ios functions, specifically imbue(), copyfmt(), and ~ios().

Definition at line 504 of file ios_base.h.

5.621.2.3 `typedef _Ios_Fmtflags std::ios_base::fmtflags` `[inherited]`

This is a bitmask type.

`_Ios_Fmtflags` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `fmtflags` are:

- `boolalpha`
- `dec`
- `fixed`
- `hex`
- `internal`
- `left`
- `oct`
- `right`
- `scientific`
- `showbase`
- `showpoint`
- `showpos`
- `skipws`
- `unitbuf`
- `uppercase`
- `adjustfield`
- `basefield`
- `floatfield`

Definition at line 323 of file ios_base.h.

5.621.2.4 typedef _Ios_Iostate std::ios_base::iostate [inherited]

This is a bitmask type.

_Ios_Iostate is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type iostate are:

- badbit
- eofbit
- failbit
- goodbit

Definition at line 398 of file ios_base.h.

5.621.2.5 typedef _Ios_Openmode std::ios_base::openmode [inherited]

This is a bitmask type.

_Ios_Openmode is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type openmode are:

- app
- ate
- binary
- in
- out
- trunc

Definition at line 429 of file ios_base.h.

5.621.2.6 typedef _Ios_Seekdir std::ios_base::seekdir [inherited]

This is an enumerated type.

_Ios_Seekdir is implementation-defined. Defined values of type seekdir are:

- beg
- cur, equivalent to `SEEK_CUR` in the C standard library.
- end, equivalent to `SEEK_END` in the C standard library.

Definition at line 461 of file ios_base.h.

5.621.3 Member Enumeration Documentation

5.621.3.1 enum `std::ios_base::event` `[inherited]`

The set of events that may be passed to an event callback.

`erase_event` is used during `~ios()` and `copyfmt()`. `imbue_event` is used during `imbue()`. `copyfmt_event` is used during `copyfmt()`.

Definition at line 487 of file `ios_base.h`.

5.621.4 Constructor & Destructor Documentation

5.621.4.1 `template<typename _CharT, typename _Traits, typename _Alloc> std::basic_ostringstream<_CharT, _Traits, _Alloc>::basic_ostringstream (ios_base::openmode __mode = ios_base::out)` `[inline], [explicit]`

Default constructor starts with an empty string buffer.

Parameters

<code>__mode</code>	Whether the buffer can read, or write, or both.
---------------------	---

`ios_base::out` is automatically included in `mode`.

Initializes `sb` using `mode|out`, and passes `&sb` to the base class initializer. Does not allocate any buffer.

That's a lie. We initialize the base class with `NULL`, because the string class does its own memory management.

Definition at line 547 of file `sstream`.

5.621.4.2 `template<typename _CharT, typename _Traits, typename _Alloc> std::basic_ostringstream<_CharT, _Traits, _Alloc>::basic_ostringstream (const __string_type & __str, ios_base::openmode __mode = ios_base::out)` `[inline], [explicit]`

Starts with an existing string buffer.

Parameters

<code>__str</code>	A string to copy as a starting buffer.
<code>__mode</code>	Whether the buffer can read, or write, or both.

`ios_base::out` is automatically included in `mode`.

Initializes `sb` using `str` and `mode|out`, and passes `&sb` to the base class initializer.

That's a lie. We initialize the base class with `NULL`, because the string class does its own memory management.

Definition at line 565 of file `sstream`.

5.621.4.3 `template<typename _CharT, typename _Traits, typename _Alloc> std::basic_ostringstream< _CharT, _Traits, _Alloc >::~~basic_ostringstream () [inline]`

The destructor does nothing.

The buffer is deallocated by the stringbuf object, not the formatting stream.

Definition at line 576 of file sstream.

5.621.5 Member Function Documentation

5.621.5.1 `const locale& std::ios_base::_M_getloc () const [inline],[inherited]`

Locale access.

Returns

A reference to the current locale.

Like getloc above, but returns a reference instead of generating a copy.

Definition at line 774 of file ios_base.h.

Referenced by `std::time_get< _CharT, _Inlter >::do_date_order()`, `std::time_get< _CharT, _Inlter >::do_get()`, `std::money_get< _CharT, _Inlter >::do_get()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::time_get< _CharT, _Inlter >::do_get_date()`, `std::time_get< _CharT, _Inlter >::do_get_monthname()`, `std::time_get< _CharT, _Inlter >::do_get_time()`, `std::time_get< _CharT, _Inlter >::do_get_weekday()`, `std::time_get< _CharT, _Inlter >::do_get_year()`, `std::time_put< _CharT, _Outlter >::do_put()`, `std::num_put< _CharT, _Outlter >::do_put()`, `std::time_get< _CharT, _Inlter >::get()`, and `std::time_put< _CharT, _Outlter >::put()`.

5.621.5.2 `template<typename _CharT, typename _Traits> void std::basic_ostream< _CharT, _Traits >::_M_write (const char_type * __s, streamsize __n) [inline],[inherited]`

Core write functionality, without sentry.

Parameters

<code>__s</code>	The array to insert.
<code>__n</code>	Maximum number of characters to insert.

Definition at line 311 of file ostream.

Referenced by `std::basic_ostream< _CharT, _Traits >::write()`.

5.621.5.3 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::bad () const [inline],[inherited]`

Fast error checking.

Returns

True if the badbit is set.

Note that other iostate flags may also be set.

Definition at line 211 of file `basic_ios.h`.

5.621.5.4 `template<typename _CharT, typename _Traits> void std::basic_ios<_CharT, _Traits>::clear (iostate __state = goodbit)` `[inherited]`

[Re]sets the error state.

Parameters

<code>__state</code>	The new state flag(s) to set.
----------------------	-------------------------------

See `std::ios_base::iostate` for the possible bit values. Most users will not need to pass an argument.

Definition at line 41 of file `basic_ios.tcc`.

Referenced by `std::basic_ios< char, _Traits>::exceptions()`, `std::__detail::operator>>()`, `std::basic_istream< _CharT, _Traits>::putback()`, `std::basic_ios< char, _Traits>::rdstate()`, `std::basic_istream< _CharT, _Traits>::seekg()`, `std::basic_ios< char, _Traits>::setstate()`, and `std::basic_istream< _CharT, _Traits>::unget()`.

5.621.5.5 `template<typename _CharT, typename _Traits> basic_ios<_CharT, _Traits> & std::basic_ios<_CharT, _Traits>::copyfmt (const basic_ios<_CharT, _Traits> & __rhs)` `[inherited]`

Copies fields of `__rhs` into this.

Parameters

<code>__rhs</code>	The source values for the copies.
--------------------	-----------------------------------

Returns

Reference to this object.

All fields of `__rhs` are copied into this object except that `rdbuf()` and `rdstate()` remain unchanged. All values in the `pword` and `iword` arrays are copied. Before copying, each callback is invoked with `erase_event`. After copying, each (new) callback is invoked with `copyfmt_event`. The final step is to copy `exceptions()`.

Definition at line 63 of file `basic_ios.tcc`.

References `std::basic_ios< _CharT, _Traits>::exceptions()`, `std::basic_ios< _CharT, _Traits>::fill()`, `std::ios_base::flags()`, `std::ios_base::getloc()`, `std::ios_base::precision()`, `std::basic_ios< _CharT, _Traits>::tie()`, `std::tie()`, and `std::ios_base::width()`.

Referenced by `std::basic_ios< char, _Traits>::rdbuf()`.

5.621.5.6 `template<typename _CharT, typename _Traits> bool std::basic_ios<_CharT, _Traits>::eof () const`
`[inline], [inherited]`

Fast error checking.

Returns

True if the eofbit is set.

Note that other iostate flags may also be set.

Definition at line 190 of file basic_ios.h.

5.621.5.7 `template<typename _CharT, typename _Traits> iostate std::basic_ios<_CharT, _Traits>::exceptions () const`
`[inline], [inherited]`

Throwing exceptions on errors.

Returns

The current exceptions mask.

This changes nothing in the stream. See the one-argument version of exceptions(iostate) for the meaning of the return value.

Definition at line 222 of file basic_ios.h.

Referenced by std::basic_ios<_CharT, _Traits>::copyfmt(), and std::basic_ios<char, _Traits>::setstate().

5.621.5.8 `template<typename _CharT, typename _Traits> void std::basic_ios<_CharT, _Traits>::exceptions (iostate __except)`
`[inline], [inherited]`

Throwing exceptions on errors.

Parameters

<code>__except</code>	The new exceptions mask.
-----------------------	--------------------------

By default, error flags are set silently. You can set an exceptions mask for each stream; if a bit in the mask becomes set in the error flags, then an exception of type std::ios_base::failure is thrown.

If the error flag is already set when the exceptions mask is added, the exception is immediately thrown. Try running the following under GCC 3.1 or later:

```
1 #include <iostream>
2 #include <fstream>
3 #include <exception>
4
5 int main()
```

```

6 {
7     std::set_terminate (__gnu_cxx::__verbose_terminate_handler);
8
9     std::ifstream f ("/etc/motd");
10
11     std::cerr << "Setting badbit\n";
12     f.setstate (std::ios_base::badbit);
13
14     std::cerr << "Setting exception mask\n";
15     f.exceptions (std::ios_base::badbit);
16 }

```

Definition at line 257 of file `basic_ios.h`.

5.621.5.9 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::fail () const`
`[inline], [inherited]`

Fast error checking.

Returns

True if either the badbit or the failbit is set.

Checking the badbit in `fail()` is historical practice. Note that other iostate flags may also be set.

Definition at line 201 of file `basic_ios.h`.

Referenced by `std::basic_ios< char, _Traits >::operator bool()`, `std::basic_ios< char, _Traits >::operator!()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, and `std::regex_traits< _Ch_type >::value()`.

5.621.5.10 `template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::fill () const`
`[inline], [inherited]`

Retrieves the *empty* character.

Returns

The current fill character.

It defaults to a space (' ') in the current locale.

Definition at line 370 of file `basic_ios.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::basic_ios< char, _Traits >::fill()`, `std::normal_distribution< _RealType >::operator()()`, `std::gamma_distribution< _RealType >::operator()()`, `std::negative_binomial_distribution< _IntType >::operator()()`, and `std::operator>>()`.

5.621.5.11 `template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::fill (char_type __ch`
`) [inline], [inherited]`

Sets a new *empty* character.

Parameters

<code>__ch</code>	The new character.
-------------------	--------------------

Returns

The previous fill character.

The fill character is used to fill out space when P+ characters have been requested (e.g., via `setw`), Q characters are actually used, and Q<P. It defaults to a space (' ') in the current locale.

Definition at line 390 of file `basic_ios.h`.

5.621.5.12 `fmtflags std::ios_base::flags() const` `[inline]`, `[inherited]`

Access to format flags.

Returns

The format control flags for both input and output.

Definition at line 619 of file `ios_base.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >::discard()`, `std::money_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::operator()`, `std::discard_block_engine< _RandomNumberEngine, __p, __r >::operator()`, `std::shuffle_order_engine< _RandomNumberEngine, __k >::operator()`, `std::normal_distribution< _RealType >::operator()`, `std::gamma_distribution< _RealType >::operator()`, `std::binomial_distribution< _IntType >::operator()`, `std::negative_binomial_distribution< _IntType >::operator()`, `std::poisson_distribution< _IntType >::operator()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, `std::__detail::operator>>()`, `std::operator>>()`, `std::linear_congruential_engine< _UIntType, __a, __c, __m >::seed()`, and `std::basic_istream< _CharT, _Traits >::sentry::sentry()`.

5.621.5.13 `fmtflags std::ios_base::flags(fmtflags __fmtfl)` `[inline]`, `[inherited]`

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

Definition at line 630 of file `ios_base.h`.

5.621.5.14 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::flush () [inherited]`

Synchronizing the stream buffer.

Returns

*this

If `rdbuf ()` is a null pointer, changes nothing.

Otherwise, calls `rdbuf ()->pubsync ()`, and if that returns -1, sets `badbit`.

Definition at line 211 of file `ostream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, `std::basic_ios< _CharT, _Traits>::setstate()`, and `std::basic_ostream< _CharT, _Traits>::tellp()`.

Referenced by `std::operator<<()`, and `std::basic_ostream< _CharT, _Traits>::write()`.

5.621.5.15 `locale std::ios_base::getloc () const [inline], [inherited]`

Locale access.

Returns

A copy of the current locale.

If `imbue (loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale ()`, the global C++ locale.

Definition at line 763 of file `ios_base.h`.

Referenced by `std::basic_ios< _CharT, _Traits>::copyfmt()`, `std::money_put< _CharT, _Outlter>::do_put()`, `std::operator>>()`, and `std::ws()`.

5.621.5.16 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits>::good () const [inline], [inherited]`

Fast error checking.

Returns

True if no error flags are set.

A wrapper around `rdstate`.

Definition at line 180 of file `basic_ios.h`.

Referenced by `std::__detail::operator>>()`, `std::basic_ostream< _CharT, _Traits>::sentry::sentry()`, and `std::basic_istream< _CharT, _Traits>::sentry::sentry()`.

5.621.5.17 `template<typename _CharT, typename _Traits> locale std::basic_ios< _CharT, _Traits>::imbue (const locale & __loc) [inherited]`

Moves to a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls `ios_base::imbue(loc)`, and if a stream buffer is associated with this stream, calls that buffer's `pubimbue(loc)`.

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>.

Definition at line 114 of file `basic_ios.tcc`.

References `std::ios_base::imbue()`.

Referenced by `std::basic_ios<char, _Traits>::fill()`.

5.621.5.18 `template<typename _CharT, typename _Traits> void std::basic_ios<_CharT, _Traits>::init(basic_streambuf<_CharT, _Traits> * __sb)` `[protected]`, `[inherited]`

All setup is performed here.

This is called from the public constructor. It is not virtual and cannot be redefined.

Definition at line 126 of file `basic_ios.tcc`.

Referenced by `std::basic_ios<char, _Traits>::basic_ios()`.

5.621.5.19 `long& std::ios_base::iword(int __ix)` `[inline]`, `[inherited]`

Access to integer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to an integer associated with the index.

The `iword` function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 809 of file `ios_base.h`.

5.621.5.20 `template<typename _CharT, typename _Traits> char std::basic_ios<_CharT, _Traits>::narrow (char_type __c, char __dfault) const [inline], [inherited]`

Squeezes characters.

Parameters

<code>__c</code>	The character to narrow.
<code>__dfault</code>	The character to narrow.

Returns

The narrowed character.

Maps a character of `char_type` to a character of `char`, if possible.

Returns the result of

```
1 std::use_facet<ctype<char_type> > (getloc()) .narrow(c, dfault)
```

Additional notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>.

Definition at line 430 of file `basic_ios.h`.

5.621.5.21 `template<typename _CharT, typename _Traits> std::basic_ios<_CharT, _Traits>::operator bool () const [inline], [explicit], [inherited]`

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`.

Definition at line 117 of file `basic_ios.h`.

5.621.5.22 `template<typename _CharT, typename _Traits> bool std::basic_ios<_CharT, _Traits>::operator! () const [inline], [inherited]`

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`.

Definition at line 125 of file `basic_ios.h`.

5.621.5.23 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits >::operator<<(__ostream_type &(*)(__ostream_type &)__pf) [inline], [inherited]`

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

Definition at line 108 of file `ostream`.

Referenced by `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, and `std::basic_ostream< ↵ _CharT, _Traits >::sentry::sentry()`.

5.621.5.24 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits >::operator<<(__ios_type &(*)(__ios_type &)__pf) [inline], [inherited]`

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

Definition at line 117 of file `ostream`.

5.621.5.25 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits >::operator<<(ios_base &(*)(ios_base &)__pf) [inline], [inherited]`

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

Definition at line 127 of file `ostream`.

5.621.5.26 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits >::operator<<(long __n) [inline], [inherited]`

Integer arithmetic inserters.

Parameters

<code>↵</code>	A variable of builtin integral type.
<code>__n</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 166 of file `ostream`.

5.621.5.27 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(unsigned long __n) [inline],[inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 170 of file `ostream`.

5.621.5.28 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(bool __n) [inline],[inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 174 of file `ostream`.

5.621.5.29 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::operator<<(short __n) [inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 92 of file `ostream.tcc`.

References `std::ios_base::basefield`, `std::ios_base::flags()`, `std::ios_base::hex`, `std::ios_base::oct`, and `std::basic_ostream<_CharT, _Traits>::operator<<()`.

5.621.5.30 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(unsigned short __n) [inline], [inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 181 of file `ostream`.

5.621.5.31 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::operator<<(int __n) [inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 106 of file `ostream.tcc`.

References `std::ios_base::basefield`, `std::ios_base::flags()`, `std::ios_base::hex`, `std::ios_base::oct`, and `std::basic_ostream<_CharT, _Traits>::operator<<()`.

5.621.5.32 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(unsigned int __n) [inline],[inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 192 of file `ostream`.

5.621.5.33 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(long long __n) [inline],[inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 201 of file `ostream`.

5.621.5.34 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(unsigned long long __n) [inline],[inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 205 of file ostream.

5.621.5.35 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(double __f) [inline], [inherited]`

Floating point arithmetic inserters.

Parameters

↩	A variable of builtin floating point type.
↩	
↩	
↩	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 220 of file ostream.

5.621.5.36 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(float __f) [inline], [inherited]`

Floating point arithmetic inserters.

Parameters

↩	A variable of builtin floating point type.
↩	
↩	
↩	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 224 of file ostream.

5.621.5.37 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(long double __f) [inline],[inherited]`

Floating point arithmetic inserters.

Parameters

<code>↔</code>	A variable of builtin floating point type.
<code>__↔</code>	
<code>↔</code>	
<code>__↔</code>	
<code>f</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 232 of file `ostream`.

5.621.5.38 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(const void * __p) [inline],[inherited]`

Pointer arithmetic inserters.

Parameters

<code>__↔</code>	A variable of pointer type.
<code>__p</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 245 of file `ostream`.

5.621.5.39 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::operator<<(__streambuf_type * __sb) [inherited]`

Extracting from another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is `NULL`, the stream will set `failbit` in its error state.

Characters are extracted from `__sb` and inserted into `*this` until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output sequence fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs while getting a character from `__sb`, which sets `failbit` in the error state

If the function inserts no characters, `failbit` is set.

Definition at line 120 of file `ostream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

5.621.5.40 streamsize std::ios_base::precision () const [inline],[inherited]

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Definition at line 689 of file `ios_base.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::normal_distribution< _RealType >::operator()()`, `std::gamma_distribution< _RealType >::operator()()`, `std::negative_binomial_distribution< _IntType >::operator()()`, and `std::operator>>()`.

5.621.5.41 streamsize std::ios_base::precision (streamsize __prec) [inline],[inherited]

Changing flags.

Parameters

<code>__prec</code>	The new precision value.
---------------------	--------------------------

Returns

The previous value of `precision()`.

Definition at line 698 of file `ios_base.h`.

5.621.5.42 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::put (char_type __c) [inherited]`

Simple insertion.

Parameters

<code>__c</code>	The character to insert.
------------------	--------------------------

Returns

`*this`

Tries to insert `__c`.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 149 of file ostream.tcc.

References `std::ios_base::badbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, `std::basic_ios< _CharT, _Traits>::setstate()`, and `std::basic_ostream< _CharT, _Traits>::write()`.

Referenced by `std::basic_ostream< _CharT, _Traits>::operator<<()`.

5.621.5.43 `void*& std::ios_base::pword (int __ix) [inline],[inherited]`

Access to void pointer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to a `void*` associated with the index.

The `pword` function provides access to an array of pointers that can be used for any purpose. The array grows as required to hold the supplied index. All pointers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 830 of file ios_base.h.

5.621.5.44 `template<typename _CharT, typename _Traits> basic_streambuf< _CharT, _Traits > * std::basic_ios< _CharT, _Traits >::rdbuf (basic_streambuf< _CharT, _Traits > * __sb)` [inherited]

Changing the underlying buffer.

Parameters

<code>__sb</code>	The new stream buffer.
-------------------	------------------------

Returns

The previous stream buffer.

Associates a new buffer with the current stream, and clears the error state.

Due to historical accidents which the LWG refuses to correct, the I/O library suffers from a design error: this function is hidden in derived classes by overrides of the zero-argument `rdbuf()`, which is non-virtual for hysterical raisins. As a result, you must use explicit qualifications to access this function via any derived class. For example:

```
1 std::fstream    foo;           // or some other derived type
2 std::streambuf* p = .....;
3
4 foo.ios::rdbuf(p);           // ios == basic_ios<char>
```

Definition at line 53 of file `basic_ios.tcc`.

5.621.5.45 `template<typename _CharT, typename _Traits, typename _Alloc> __stringbuf_type* std::basic_ostringstream< _CharT, _Traits, _Alloc >::rdbuf () const` [inline]

Accessing the underlying buffer.

Returns

The current `basic_stringbuf` buffer.

This hides both signatures of `std::basic_ios::rdbuf()`.

Definition at line 616 of file `sstream`.

5.621.5.46 `template<typename _CharT, typename _Traits> iostate std::basic_ios< _CharT, _Traits >::rdstate () const` [inline], [inherited]

Returns the error state of the stream buffer.

Returns

A bit pattern (well, isn't everything?)

See `std::ios_base::iostate` for the possible bit values. Most users will call one of the interpreting wrappers, e.g., `good()`.

Definition at line 137 of file `basic_ios.h`.

Referenced by `std::basic_ios< char, _Traits >::bad()`, `std::basic_ios< char, _Traits >::eof()`, `std::basic_ios< char, _Traits >::fail()`, `std::basic_ios< char, _Traits >::good()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ios< char, _Traits >::setstate()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

5.621.5.47 `void std::ios_base::register_callback (event_callback __fn, int __index)` [inherited]

Add the callback `__fn` with parameter `__index`.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

5.621.5.48 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::seekp (pos_type __pos) [inherited]`

Changing the current write position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

`*this`

If `fail()` is not true, calls `rdbuf()->pubseekpos(pos)`. If that function fails, sets failbit.

Definition at line 258 of file `ostream.tcc`.

References `std::ios_base::badbit`, `std::basic_ios< _CharT, _Traits>::fail()`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::ios_base::out`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

Referenced by `std::basic_ostream< _CharT, _Traits>::tellp()`.

5.621.5.49 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::seekp (off_type __off, ios_base::seekdir __dir) [inherited]`

Changing the current write position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

`*this`

If `fail()` is not true, calls `rdbuf()->pubseekoff(off, dir)`. If that function fails, sets failbit.

Definition at line 290 of file `ostream.tcc`.

References std::ios_base::badbit, std::basic_ios< _CharT, _Traits >::fail(), std::ios_base::failbit, std::ios_base::goodbit, std::ios_base::out, std::basic_ios< _CharT, _Traits >::rdbuf(), and std::basic_ios< _CharT, _Traits >::setstate().

5.621.5.50 **fmtflags** std::ios_base::setf (**fmtflags** *__fmtfl*) [inline], [inherited]

Setting new format flags.

Parameters

<i>__fmtfl</i>	Additional flags to set.
----------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set.

Definition at line 646 of file ios_base.h.

Referenced by std::boolalpha(), std::dec(), std::fixed(), std::hex(), std::hexfloat(), std::left(), std::oct(), std::__detail<::operator>>(), std::right(), std::scientific(), std::showbase(), std::showpoint(), std::showpos(), std::skipws(), std::unitbuf(), and std::uppercase().

5.621.5.51 **fmtflags** std::ios_base::setf (**fmtflags** *__fmtfl*, **fmtflags** *__mask*) [inline], [inherited]

Setting new format flags.

Parameters

<i>__fmtfl</i>	Additional flags to set.
<i>__mask</i>	The flags mask for <i>__fmtfl</i> .

Returns

The previous format control flags.

This function clears *mask* in the format flags, then sets *__fmtfl* & *mask*. An example mask is ios_base::adjustfield.

Definition at line 663 of file ios_base.h.

5.621.5.52 **template**<typename *_CharT*, typename *_Traits*> void std::basic_ios< *_CharT*, *_Traits* >::setstate (**iosstate** *__state*) [inline], [inherited]

Sets additional flags in the error state.

Parameters

<code>__state</code>	The additional state flag(s) to set.
----------------------	--------------------------------------

See `std::ios_base::iostate` for the possible bit values.

Definition at line 157 of file `basic_ios.h`.

Referenced by `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::tr2::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::unset()`, and `std::ws()`.

5.621.5.53 `template<typename _CharT, typename _Traits, typename _Alloc> __string_type std::basic_ostringstream< _CharT, _Traits, _Alloc>::str () const [inline]`

Copying out the string buffer.

Returns

`rdbuf () -> str ()`

Definition at line 624 of file `sstream`.

Referenced by `std::__detail::operator<<()`.

5.621.5.54 `template<typename _CharT, typename _Traits, typename _Alloc> void std::basic_ostringstream< _CharT, _Traits, _Alloc>::str (const __string_type & __s) [inline]`

Setting a new buffer.

Parameters

<code>__s</code>	The string to use as a new sequence.
------------------	--------------------------------------

Calls `rdbuf () -> str (s)`.

Definition at line 634 of file `sstream`.

5.621.5.55 `static bool std::ios_base::sync_with_stdio (bool __sync = true) [static], [inherited]`

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., stdout) and the standard C++ objects (e.g., cout). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>

5.621.5.56 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>::pos_type
std::basic_ostream<_CharT, _Traits>::tellp () [inherited]`

Getting the current write position.

Returns

A file position object.

If `fail()` is not false, returns `pos_type(-1)` to indicate failure. Otherwise returns `rdbuf() -> pubseekoff(0, cur, out)`.

Definition at line 237 of file `ostream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::cur`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::ios_base::out`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ostream<_CharT, _Traits>::seekp()`.

Referenced by `std::basic_ostream<_CharT, _Traits>::flush()`.

5.621.5.57 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>* std::basic_ios<_CharT,
_Traits>::tie () const [inline], [inherited]`

Fetches the current *tied* stream.

Returns

A pointer to the tied stream, or NULL if the stream is not tied.

A stream may be *tied* (or synchronized) to a second output stream. When this stream performs any I/O, the tied stream is first flushed. For example, `std::cin` is tied to `std::cout`.

Definition at line 295 of file `basic_ios.h`.

Referenced by `std::basic_ios<char, _Traits>::basic_ios()`, `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`, and `std::basic_istream<_CharT, _Traits>::sentry::sentry()`.

5.621.5.58 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>* std::basic_ios<_CharT,
_Traits>::tie (basic_ostream<_CharT, _Traits>* __tiestr) [inline], [inherited]`

Ties this stream to an output stream.

Parameters

<code>__tiestr</code>	The output stream.
-----------------------	--------------------

Returns

The previously tied output stream, or NULL if the stream was not tied.

This sets up a new tie; see `tie()` for more.

Definition at line 307 of file `basic_ios.h`.

5.621.5.59 `void std::ios_base::unsetf (fmtflags __mask)` `[inline]`, `[inherited]`

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

Definition at line 678 of file `ios_base.h`.

Referenced by `std::defaultfloat()`, `std::noboolalpha()`, `std::noshowbase()`, `std::noshowpoint()`, `std::noshowpos()`, `std::noskipws()`, `std::nounitbuf()`, and `std::nouppercase()`.

5.621.5.60 `template<typename _CharT, typename _Traits> char_type std::basic_ios<_CharT, _Traits>::widen (char __c)`
`const` `[inline]`, `[inherited]`

Widens characters.

Parameters

<code>__c</code>	The character to widen.
------------------	-------------------------

Returns

The widened character.

Maps a character of `char` to a character of `char_type`.

Returns the result of

```
1 std::use_facet<ctype<char_type>> >(getloc()).widen(c)
```

Additional l10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Definition at line 449 of file basic_ios.h.

Referenced by std::basic_ios< char, _Traits >::fill(), std::getline(), std::normal_distribution< _RealType >::operator>(), std::gamma_distribution< _RealType >::operator(), std::negative_binomial_distribution< _IntType >::operator(), std::tr2::operator>>(), and std::operator>>().

5.621.5.61 streamsize std::ios_base::width () const [inline], [inherited]

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Definition at line 712 of file ios_base.h.

Referenced by std::basic_ios< _CharT, _Traits >::copyfmt(), std::money_get< _CharT, _InIter >::do_get(), std::num_get< _CharT, _InIter >::do_get(), std::num_put< _CharT, _OutIter >::do_put(), and std::operator>>().

5.621.5.62 streamsize std::ios_base::width (streamsize __wide) [inline], [inherited]

Changing flags.

Parameters

<code>__wide</code>	The new width value.
---------------------	----------------------

Returns

The previous value of width().

Definition at line 721 of file ios_base.h.

5.621.5.63 template<typename _CharT, typename _Traits > basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::write (const char_type * __s, streamsize __n) [inherited]

Character string insertion.

Parameters

<code>__s</code>	The array to insert.
<code>__n</code>	Maximum number of characters to insert.

Returns

*this

Characters are copied from `___s` and inserted into the stream until one of the following happens:

- `___n` characters are inserted
- inserting into the output sequence fails (in this case, `badbit` will be set in the stream's error state)

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 183 of file `ostream.tcc`.

References `std::basic_ostream< _CharT, _Traits >::_M_write()`, `std::ios_base::badbit`, and `std::basic_ostream< _CharT, _Traits >::flush()`.

Referenced by `std::basic_ostream< _CharT, _Traits >::put()`.

5.621.5.64 `static int std::ios_base::xalloc () throw` `[static]`, `[inherited]`

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the `iword` and `pword` functions. The expectation is that an application calls `xalloc` in order to obtain an index in the `iword` and `pword` arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. `xalloc` is guaranteed to return an index that is safe to use in the `iword` and `pword` arrays.

5.621.6 Member Data Documentation

5.621.6.1 `const fmtflags std::ios_base::adjustfield` `[static]`, `[inherited]`

A mask of `left|right|internal`. Useful for the 2-arg form of `setf`.

Definition at line 378 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::internal()`, `std::left()`, and `std::right()`.

5.621.6.2 const openmode std::ios_base::app [static],[inherited]

Seek to end before each write.

Definition at line 432 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::overflow(), and std::basic_filebuf< _CharT, _Traits >::xsputn().

5.621.6.3 const openmode std::ios_base::ate [static],[inherited]

Open and seek to end immediately after opening.

Definition at line 435 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::open().

5.621.6.4 const iostate std::ios_base::badbit [static],[inherited]

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Definition at line 402 of file ios_base.h.

Referenced by std::basic_ios< char, _Traits >::bad(), std::basic_ios< char, _Traits >::fail(), std::basic_ostream< ↵
_CharT, _Traits >::flush(), std::basic_istream< _CharT, _Traits >::get(), std::basic_istream< _CharT, _Traits >↵
::getline(), std::basic_istream< _CharT, _Traits >::ignore(), std::basic_ostream< _CharT, _Traits >::operator<<(),
std::operator<<(), std::basic_istream< _CharT, _Traits >::operator>>(), std::operator>>(), std::basic_istream< ↵
_CharT, _Traits >::peek(), std::basic_ostream< _CharT, _Traits >::put(), std::basic_istream< _CharT, _Traits >↵
::putback(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::readsome(), std↵
::basic_istream< _CharT, _Traits >::seekg(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_ostream<
_CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< ↵
_CharT, _Traits >::sync(), std::basic_istream< _CharT, _Traits >::tellg(), std::basic_ostream< _CharT, _Traits >::tellp(),
std::basic_istream< _CharT, _Traits >::unget(), and std::basic_ostream< _CharT, _Traits >::write().

5.621.6.5 const fmtflags std::ios_base::basefield [static],[inherited]

A mask of dec|oct|hex. Useful for the 2-arg form of setf.

Definition at line 381 of file ios_base.h.

Referenced by std::dec(), std::num_get< _CharT, _InIter >::do_get(), std::num_put< _CharT, _OutIter >::do_put(),
std::hex(), std::oct(), and std::basic_ostream< _CharT, _Traits >::operator<<().

5.621.6.6 const seekdir std::ios_base::beg [static],[inherited]

Request a seek relative to the beginning of the stream.

Definition at line 464 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::seekpos(), and __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits
>::sync().

5.621.6.7 `const openmode std::ios_base::binary` `[static],[inherited]`

Perform input and output in binary mode (as opposed to text mode). This is probably not what you think it is; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.binary>.

Definition at line 440 of file `ios_base.h`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::showmanyc()`.

5.621.6.8 `const fmtflags std::ios_base::boolalpha` `[static],[inherited]`

Insert/extract `bool` in alphabetic rather than numeric format.

Definition at line 326 of file `ios_base.h`.

Referenced by `std::boolalpha()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_put<_CharT, _OutIter>::do_put()`, and `std::noboolalpha()`.

5.621.6.9 `const seekdir std::ios_base::cur` `[static],[inherited]`

Request a seek relative to the current position within the sequence.

Definition at line 467 of file `ios_base.h`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::imbue()`, `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_filebuf<_CharT, _Traits>::pbackfail()`, `std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekoff()`, `std::basic_filebuf<_CharT, _Traits>::seekoff()`, `__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>::sync()`, `std::basic_istream<_CharT, _Traits>::tellg()`, and `std::basic_ostream<_CharT, _Traits>::tellp()`.

5.621.6.10 `const fmtflags std::ios_base::dec` `[static],[inherited]`

Converts integer input or generates integer output in decimal base.

Definition at line 329 of file `ios_base.h`.

Referenced by `std::dec()`.

5.621.6.11 `const seekdir std::ios_base::end` `[static],[inherited]`

Request a seek relative to the current end of the sequence.

Definition at line 470 of file `ios_base.h`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::open()`, and `std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekoff()`.

5.621.6.12 const iostate std::ios_base::eofbit [static],[inherited]

Indicates that an input operation reached the end of an input sequence.

Definition at line 405 of file ios_base.h.

Referenced by std::time_get< _CharT, _InIter >::do_get(), std::num_get< _CharT, _InIter >::do_get(), std::time_get< _CharT, _InIter >::do_get_date(), std::time_get< _CharT, _InIter >::do_get_monthname(), std::time_get< _CharT, _InIter >::do_get_time(), std::time_get< _CharT, _InIter >::do_get_weekday(), std::time_get< _CharT, _InIter >::do_get_year(), std::basic_ios< char, _Traits >::eof(), std::basic_istream< _CharT, _Traits >::get(), std::time_get< _CharT, _InIter >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::basic_istream< _CharT, _Traits >::ignore(), std::basic_istream< _CharT, _Traits >::operator>>(), std::operator>>(), std::basic_istream< _CharT, _Traits >::peek(), std::basic_istream< _CharT, _Traits >::putback(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::readsome(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_istream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::unget(), and std::ws().

5.621.6.13 const iostate std::ios_base::failbit [static],[inherited]

Indicates that an input operation failed to read the expected characters, or that an output operation failed to generate the desired characters.

Definition at line 410 of file ios_base.h.

Referenced by std::time_get< _CharT, _InIter >::do_date_order(), std::num_get< _CharT, _InIter >::do_get(), std::time_get< _CharT, _InIter >::do_get_monthname(), std::time_get< _CharT, _InIter >::do_get_weekday(), std::time_get< _CharT, _InIter >::do_get_year(), std::basic_ios< char, _Traits >::fail(), std::basic_istream< _CharT, _Traits >::get(), std::time_get< _CharT, _InIter >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::basic_ostream< _CharT, _Traits >::operator<<(), std::basic_istream< _CharT, _Traits >::operator>>(), std::operator>>(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_ostream< _CharT, _Traits >::sentry::sentry(), and std::basic_istream< _CharT, _Traits >::sentry::sentry().

5.621.6.14 const fmtflags std::ios_base::fixed [static],[inherited]

Generate floating-point output in fixed-point notation.

Definition at line 332 of file ios_base.h.

Referenced by std::num_get< _CharT, _InIter >::do_get(), std::fixed(), and std::hexfloat().

5.621.6.15 const fmtflags std::ios_base::floatfield [static],[inherited]

A mask of scientific|fixed. Useful for the 2-arg form of setf.

Definition at line 384 of file ios_base.h.

Referenced by std::defaultfloat(), std::num_get< _CharT, _InIter >::do_get(), std::fixed(), std::hexfloat(), and std::scientific().

5.621.6.16 `const iostate std::ios_base::goodbit` `[static], [inherited]`

Indicates all is well.

Definition at line 413 of file `ios_base.h`.

Referenced by `std::time_get< _CharT, _InIter >::do_date_order()`, `std::time_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::time_get< _CharT, _InIter >::do_get_monthname()`, `std::time_get< _CharT, _InIter >::do_get_weekday()`, `std::time_get< _CharT, _InIter >::do_get_year()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::time_get< _CharT, _InIter >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_ios< char, _Traits >::rdstate()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sync()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

5.621.6.17 `const fmtflags std::ios_base::hex` `[static], [inherited]`

Converts integer input or generates integer output in hexadecimal base.

Definition at line 335 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::hex()`, and `std::basic_ostream< _CharT, _Traits >::operator<<()`.

5.621.6.18 `const openmode std::ios_base::in` `[static], [inherited]`

Open for input. Default for `ifstream` and `fstream`.

Definition at line 443 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos()`, `std::basic_filebuf< _CharT, _Traits >::showmanyc()`, `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::underflow()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

5.621.6.19 `const fmtflags std::ios_base::internal` `[static], [inherited]`

Adds fill characters at a designated internal point in certain generated output, or identical to `right` if no such point is designated.

Definition at line 340 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, and `std::internal()`.

5.621.6.20 `const fmtflags std::ios_base::left` `[static], [inherited]`

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.)

Definition at line 344 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, and `std::left()`.

5.621.6.21 `const fmtflags std::ios_base::oct` `[static], [inherited]`

Converts integer input or generates integer output in octal base.

Definition at line 347 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _InIter >::do_get()`, `std::oct()`, and `std::basic_ostream< _CharT, _Traits >::operator<<()`.

5.621.6.22 `const openmode std::ios_base::out` `[static], [inherited]`

Open for output. Default for `ofstream` and `fstream`.

Definition at line 446 of file `ios_base.h`.

Referenced by `std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow()`, `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::pbackfail()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos()`, `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

5.621.6.23 `const fmtflags std::ios_base::right` `[static], [inherited]`

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Definition at line 351 of file `ios_base.h`.

Referenced by `std::right()`.

5.621.6.24 `const fmtflags std::ios_base::scientific` `[static], [inherited]`

Generates floating-point output in scientific notation.

Definition at line 354 of file `ios_base.h`.

Referenced by `std::hexfloat()`, and `std::scientific()`.

5.621.6.25 `const fmtflags std::ios_base::showbase` `[static], [inherited]`

Generates a prefix indicating the numeric base of generated integer output.

Definition at line 358 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::noshowbase()`, and `std::showbase()`.

5.621.6.26 `const fmtflags std::ios_base::showpoint` `[static], [inherited]`

Generates a decimal-point character unconditionally in generated floating-point output.

Definition at line 362 of file `ios_base.h`.

Referenced by `std::noshowpoint()`, and `std::showpoint()`.

5.621.6.27 `const fmtflags std::ios_base::showpos` `[static], [inherited]`

Generates a + sign in non-negative generated numeric output.

Definition at line 365 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _InIter >::do_get()`, `std::noshowpos()`, and `std::showpos()`.

5.621.6.28 `const fmtflags std::ios_base::skipws` `[static], [inherited]`

Skips leading white space before certain input operations.

Definition at line 368 of file `ios_base.h`.

Referenced by `std::noskipws()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, and `std::skipws()`.

5.621.6.29 `const openmode std::ios_base::trunc` `[static], [inherited]`

Open for input. Default for `ofstream`.

Definition at line 449 of file `ios_base.h`.

5.621.6.30 `const fmtflags std::ios_base::unitbuf` `[static], [inherited]`

Flushes output after each output operation.

Definition at line 371 of file `ios_base.h`.

Referenced by `std::nunitbuf()`, and `std::unitbuf()`.

5.621.6.31 `const fmtflags std::ios_base::uppercase` `[static], [inherited]`

Replaces certain lowercase letters with their uppercase equivalents in generated output.

Definition at line 375 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::noskipws()`, and `std::uppercase()`.

The documentation for this class was generated from the following files:

- [iosfwd](#)
- [sstream](#)

5.622 std::basic_regex< _Ch_type, _Rx_traits > Class Template Reference

Public Types

- typedef [regex_constants::syntax_option_type](#) **flag_type**
- typedef traits_type::locale_type **locale_type**
- typedef traits_type::string_type **string_type**
- typedef _Rx_traits **traits_type**
- typedef _Ch_type **value_type**

Public Member Functions

- [basic_regex](#) ()
- [basic_regex](#) (const _Ch_type * __p, [flag_type](#) __f=ECMAScript)
- [basic_regex](#) (const _Ch_type * __p, std::size_t __len, [flag_type](#) __f=ECMAScript)
- [basic_regex](#) (const [basic_regex](#) & __rhs)=default
- [basic_regex](#) ([basic_regex](#) && __rhs) noexcept=default
- template<typename _Ch_traits, typename _Ch_alloc >
 [basic_regex](#) (const [std::basic_string](#)< _Ch_type, _Ch_traits, _Ch_alloc > & __s, [flag_type](#) __f=ECMAScript)
- template<typename _FwdIter >
 [basic_regex](#) (_FwdIter __first, _FwdIter __last, [flag_type](#) __f=ECMAScript)
- [basic_regex](#) ([initializer_list](#)< _Ch_type > __l, [flag_type](#) __f=ECMAScript)
- [~basic_regex](#) ()
- [basic_regex](#) & [assign](#) (const [basic_regex](#) & __rhs)
- [basic_regex](#) & [assign](#) ([basic_regex](#) && __rhs) noexcept
- [basic_regex](#) & [assign](#) (const _Ch_type * __p, [flag_type](#) __flags=ECMAScript)
- [basic_regex](#) & [assign](#) (const _Ch_type * __p, std::size_t __len, [flag_type](#) __flags)
- template<typename _Ch_traits, typename _Alloc >
 [basic_regex](#) & [assign](#) (const [basic_string](#)< _Ch_type, _Ch_traits, _Alloc > & __s, [flag_type](#) __flags=ECMAScript)
- template<typename _InputIterator >
 [basic_regex](#) & [assign](#) (_InputIterator __first, _InputIterator __last, [flag_type](#) __flags=ECMAScript)
- [basic_regex](#) & [assign](#) ([initializer_list](#)< _Ch_type > __l, [flag_type](#) __flags=ECMAScript)
- [flag_type](#) [flags](#) () const
- [locale_type](#) [getloc](#) () const
- [locale_type](#) [imbue](#) ([locale_type](#) __loc)
- unsigned int [mark_count](#) () const
- [basic_regex](#) & [operator=](#) (const [basic_regex](#) & __rhs)
- [basic_regex](#) & [operator=](#) ([basic_regex](#) && __rhs) noexcept
- [basic_regex](#) & [operator=](#) (const _Ch_type * __p)
- [basic_regex](#) & [operator=](#) ([initializer_list](#)< _Ch_type > __l)
- template<typename _Ch_traits, typename _Alloc >
 [basic_regex](#) & [operator=](#) (const [basic_string](#)< _Ch_type, _Ch_traits, _Alloc > & __s)
- void [swap](#) ([basic_regex](#) & __rhs)

Static Public Attributes

Constants

std [28.8.1](1)

- static constexpr [flag_type](#) **icase**
- static constexpr [flag_type](#) **nosubs**
- static constexpr [flag_type](#) **optimize**
- static constexpr [flag_type](#) **collate**
- static constexpr [flag_type](#) **ECMAScript**
- static constexpr [flag_type](#) **basic**
- static constexpr [flag_type](#) **extended**
- static constexpr [flag_type](#) **awk**
- static constexpr [flag_type](#) **grep**
- static constexpr [flag_type](#) **egrep**

Friends

- template<typename _Bp, typename _Ap, typename _Cp, typename _Rp, __detail::_RegexExecutorPolicy, bool >
bool **__detail::__regex_algo_impl** (_Bp, _Bp, [match_results](#)< _Bp, _Ap > &, const [basic_regex](#)< _Cp, _Rp > &, [regex_constants::match_flag_type](#))
- template<typename, typename, typename, bool >
class **__detail::_Executor**

5.622.1 Detailed Description

```
template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>>
class std::basic_regex< _Ch_type, _Rx_traits >
```

Objects of specializations of this class represent regular expressions constructed from sequences of character type `_Ch_type`.

Storage for the regular expression is allocated and deallocated as necessary by the member functions of this class.

Definition at line 36 of file `regex.h`.

5.622.2 Constructor & Destructor Documentation

5.622.2.1 `template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> std::basic_regex< _Ch_type, _Rx_traits >::basic_regex () [inline]`

Constructs a basic regular expression that does not match any character sequence.

Definition at line 428 of file `regex.h`.

5.622.2.2 `template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> std::basic_regex< _Ch_type, _Rx_traits >::basic_regex (const _Ch_type * __p, flag_type __f = ECMAScript) [inline], [explicit]`

Constructs a basic regular expression from the sequence `[__p, __p + char_traits<_Ch_type>::length(__p))` interpreted according to the flags in `__f`.

Parameters

<code>__p</code>	A pointer to the start of a C-style null-terminated string containing a regular expression.
<code>__f</code>	Flags indicating the syntax rules and options.

Exceptions

<code>regex_error</code>	if <code>__p</code> is not a valid regular expression.
--------------------------	--

Definition at line 444 of file regex.h.

```
5.622.2.3 template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> std::basic_regex<_Ch_type,
    _Rx_traits>::basic_regex ( const _Ch_type * __p, std::size_t __len, flag_type __f = ECMAScript )
    [inline]
```

Constructs a basic regular expression from the sequence `[p, p + len)` interpreted according to the flags in `f`.

Parameters

<code>__p</code>	A pointer to the start of a string containing a regular expression.
<code>__len</code>	The length of the string containing the regular expression.
<code>__f</code>	Flags indicating the syntax rules and options.

Exceptions

<code>regex_error</code>	if <code>__p</code> is not a valid regular expression.
--------------------------	--

Definition at line 460 of file regex.h.

```
5.622.2.4 template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> std::basic_regex<_Ch_type,
    _Rx_traits>::basic_regex ( const basic_regex<_Ch_type, _Rx_traits> & __rhs ) [default]
```

Copy-constructs a basic regular expression.

Parameters

<code>__rhs</code>	A <code>regex</code> object.
--------------------	------------------------------

```
5.622.2.5 template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> std::basic_regex<_Ch_type,
    _Rx_traits>::basic_regex ( basic_regex<_Ch_type, _Rx_traits> && __rhs ) [default], [noexcept]
```

Move-constructs a basic regular expression.

Parameters

<code>__rhs</code>	A regex object.
--------------------	-----------------

5.622.2.6 `template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> template<typename _Ch_traits ,
typename _Ch_alloc > std::basic_regex<_Ch_type, _Rx_traits >::basic_regex (const std::basic_string<
_Ch_type, _Ch_traits, _Ch_alloc > &__s, flag_type __f=ECMAScript) [inline],[explicit]`

Constructs a basic regular expression from the string `s` interpreted according to the flags in `f`.

Parameters

<code>__s</code>	A string containing a regular expression.
<code>__f</code>	Flags indicating the syntax rules and options.

Exceptions

<code>regex_error</code>	if <code>__s</code> is not a valid regular expression.
--------------------------	--

Definition at line 490 of file `regex.h`.

5.622.2.7 `template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> template<typename _Fwdlter >
std::basic_regex<_Ch_type, _Rx_traits >::basic_regex (_Fwdlter __first, _Fwdlter __last, flag_type __f =
ECMAScript) [inline]`

Constructs a basic regular expression from the range `[first, last)` interpreted according to the flags in `f`.

Parameters

<code>__first</code>	The start of a range containing a valid regular expression.
<code>__last</code>	The end of a range containing a valid regular expression.
<code>__f</code>	The format flags of the regular expression.

Exceptions

<code>regex_error</code>	if <code>[__first, __last)</code> is not a valid regular expression.
--------------------------	--

Definition at line 510 of file `regex.h`.

5.622.2.8 `template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> std::basic_regex<_Ch_type,
_Rx_traits >::basic_regex (initializer_list<_Ch_type > __l, flag_type __f=ECMAScript) [inline]`

Constructs a basic regular expression from an initializer list.

Parameters

↩ _↩ ↩ _↩ /	The initializer list.
↩ _↩ ↩ _↩ f	The format flags of the regular expression.

Exceptions

<i>regex_error</i>	if <code>__l</code> is not a valid regular expression.
--------------------	--

Definition at line 523 of file regex.h.

5.622.2.9 `template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> std::basic_regex<_Ch_type, _Rx_traits>::~~basic_regex() [inline]`

Destroys a basic regular expression.

Definition at line 530 of file regex.h.

5.622.3 Member Function Documentation

5.622.3.1 `template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> basic_regex& std::basic_regex<_Ch_type, _Rx_traits>::assign(const basic_regex<_Ch_type, _Rx_traits> &__rhs) [inline]`

the real assignment operator.

Parameters

<code>__rhs</code>	Another regular expression object.
--------------------	------------------------------------

Definition at line 588 of file regex.h.

5.622.3.2 `template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> basic_regex& std::basic_regex<_Ch_type, _Rx_traits>::assign(basic_regex<_Ch_type, _Rx_traits> &&__rhs) [inline], [noexcept]`

The move-assignment operator.

Parameters

<code>__rhs</code>	Another regular expression object.
--------------------	------------------------------------

Definition at line 601 of file regex.h.

```
5.622.3.3  template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> basic_regex& std::basic_regex<
           _Ch_type, _Rx_traits >::assign ( const _Ch_type * __p, flag_type __flags = ECMAScript ) [inline]
```

Assigns a new regular expression to a regex object from a C-style null-terminated string containing a regular expression pattern.

Parameters

<code>__p</code>	A pointer to a C-style null-terminated string containing a regular expression pattern.
<code>__flags</code>	Syntax option flags.

Exceptions

<i>regex_error</i>	if <code>__p</code> does not contain a valid regular expression pattern interpreted according to <code>__flags</code> . If <code>regex_error</code> is thrown, <code>*this</code> remains unchanged.
--------------------	--

Definition at line 622 of file regex.h.

```
5.622.3.4  template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> basic_regex& std::basic_regex<
           _Ch_type, _Rx_traits >::assign ( const _Ch_type * __p, std::size_t __len, flag_type __flags ) [inline]
```

Assigns a new regular expression to a regex object from a C-style string containing a regular expression pattern.

Parameters

<code>__p</code>	A pointer to a C-style string containing a regular expression pattern.
<code>__len</code>	The length of the regular expression pattern string.
<code>__flags</code>	Syntax option flags.

Exceptions

<i>regex_error</i>	if <code>p</code> does not contain a valid regular expression pattern interpreted according to <code>__flags</code> . If <code>regex_error</code> is thrown, <code>*this</code> remains unchanged.
--------------------	--

Definition at line 639 of file regex.h.

```
5.622.3.5  template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> template<typename _Ch_traits ,
           typename _Alloc > basic_regex& std::basic_regex< _Ch_type, _Rx_traits >::assign ( const basic_string<
           _Ch_type, _Ch_traits, _Alloc > & __s, flag_type __flags = ECMAScript ) [inline]
```

Assigns a new regular expression to a regex object from a string containing a regular expression pattern.

Parameters

<code>__s</code>	A string containing a regular expression pattern.
<code>__flags</code>	Syntax option flags.

Exceptions

<i>regex_error</i>	if <code>__s</code> does not contain a valid regular expression pattern interpreted according to <code>__flags</code> . If <code>regex_error</code> is thrown, <code>*this</code> remains unchanged.
--------------------	--

Definition at line 655 of file `regex.h`.

References `std::basic_string< _CharT, _Traits, _Alloc >::data()`, and `std::basic_string< _CharT, _Traits, _Alloc >::size()`.

```
5.622.3.6 template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> template<typename _InputIterator >
    basic_regex& std::basic_regex< _Ch_type, _Rx_traits >::assign ( _InputIterator __first, _InputIterator __last,
        flag_type __flags = ECMAScript ) [inline]
```

Assigns a new regular expression to a regex object.

Parameters

<code>__first</code>	The start of a range containing a valid regular expression.
<code>__last</code>	The end of a range containing a valid regular expression.
<code>__flags</code>	Syntax option flags.

Exceptions

<i>regex_error</i>	if <code>p</code> does not contain a valid regular expression pattern interpreted according to <code>__flags</code> . If <code>regex_error</code> is thrown, the object remains unchanged.
--------------------	--

Definition at line 677 of file `regex.h`.

```
5.622.3.7 template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> basic_regex& std::basic_regex<
    _Ch_type, _Rx_traits >::assign ( initializer_list< _Ch_type > __l, flag_type __flags = ECMAScript )
    [inline]
```

Assigns a new regular expression to a regex object.

Parameters

<code>__l</code>	An initializer list representing a regular expression.
<code>__flags</code>	Syntax option flags.

Exceptions

<i>regex_error</i>	if <code>__l</code> does not contain a valid regular expression pattern interpreted according to <code>__flags</code> . If <code>regex_error</code> is thrown, the object remains unchanged.
--------------------	--

Definition at line 693 of file `regex.h`.

5.622.3.8 `template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> flag_type std::basic_regex<_Ch_type, _Rx_traits>::flags () const [inline]`

Gets the flags used to construct the regular expression or in the last call to `assign()`.

Definition at line 714 of file `regex.h`.

5.622.3.9 `template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> locale_type std::basic_regex<_Ch_type, _Rx_traits>::getloc () const [inline]`

Gets the locale currently imbued in the regular expression object.

Definition at line 736 of file `regex.h`.

5.622.3.10 `template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> locale_type std::basic_regex<_Ch_type, _Rx_traits>::imbue (locale_type __loc) [inline]`

Imbues the regular expression object with the given locale.

Parameters

<code>__loc</code>	A locale.
--------------------	-----------

Definition at line 724 of file `regex.h`.

5.622.3.11 `template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> unsigned int std::basic_regex<_Ch_type, _Rx_traits>::mark_count () const [inline]`

Gets the number of marked subexpressions within the regular expression.

Definition at line 702 of file `regex.h`.

5.622.3.12 `template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> basic_regex& std::basic_regex<_Ch_type, _Rx_traits>::operator=(const basic_regex<_Ch_type, _Rx_traits> &__rhs) [inline]`

Assigns one regular expression to another.

Definition at line 537 of file `regex.h`.

```
5.622.3.13 template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> basic_regex&
std::basic_regex<_Ch_type, _Rx_traits>::operator= ( basic_regex<_Ch_type, _Rx_traits> && __rhs )
[inline], [noexcept]
```

Move-assigns one regular expression to another.

Definition at line 544 of file regex.h.

```
5.622.3.14 template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> basic_regex&
std::basic_regex<_Ch_type, _Rx_traits>::operator= ( const _Ch_type * __p ) [inline]
```

Replaces a regular expression with a new one constructed from a C-style null-terminated string.

Parameters

<code>__p</code>	A pointer to the start of a null-terminated C-style string containing a regular expression.
------------------	---

Definition at line 555 of file regex.h.

```
5.622.3.15 template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> basic_regex&
std::basic_regex<_Ch_type, _Rx_traits>::operator= ( initializer_list<_Ch_type> __l ) [inline]
```

Replaces a regular expression with a new one constructed from an initializer list.

Parameters

<code>__l</code>	The initializer list.
------------------	-----------------------

Exceptions

<code>regex_error</code>	if <code>__l</code> is not a valid regular expression.
--------------------------	--

Definition at line 567 of file regex.h.

```
5.622.3.16 template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> template<typename _Ch_traits ,
typename _Alloc> basic_regex& std::basic_regex<_Ch_type, _Rx_traits>::operator= ( const basic_string<
_Ch_type, _Ch_traits, _Alloc> & __s ) [inline]
```

Replaces a regular expression with a new one constructed from a string.

Parameters

<code>_↔</code>	A pointer to a string containing a regular expression.
<code>_s</code>	

Definition at line 578 of file regex.h.

5.622.3.17 `template<typename _Ch_type, typename _Rx_traits = regex_traits<_Ch_type>> void std::basic_regex<_Ch_type, _Rx_traits>::swap (basic_regex<_Ch_type, _Rx_traits> &__rhs) [inline]`

Swaps the contents of two regular expression objects.

Parameters

<code>__rhs</code>	Another regular expression object.
--------------------	------------------------------------

Definition at line 746 of file regex.h.

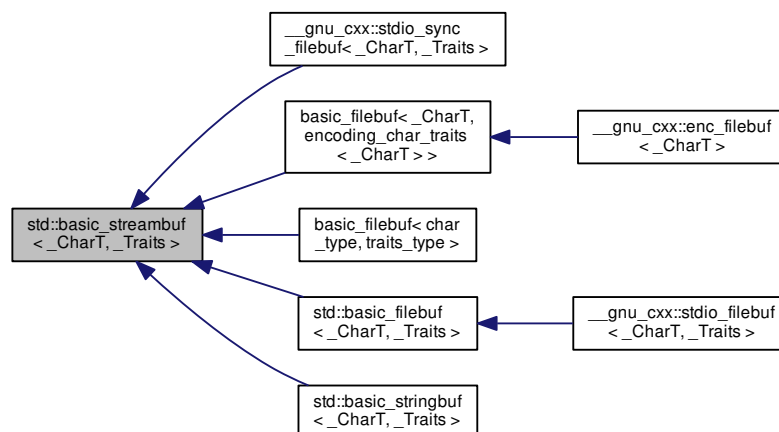
Referenced by `std::swap()`.

The documentation for this class was generated from the following file:

- [regex.h](#)

5.623 `std::basic_streambuf<_CharT, _Traits>` Class Template Reference

Inheritance diagram for `std::basic_streambuf<_CharT, _Traits>`:



Public Types

- typedef `_CharT` `char_type`
- typedef `_Traits` `traits_type`
- typedef `traits_type::int_type` `int_type`
- typedef `traits_type::pos_type` `pos_type`
- typedef `traits_type::off_type` `off_type`
- typedef `basic_streambuf< char_type, traits_type > __streambuf_type`

Public Member Functions

- virtual `~basic_streambuf()`
- `locale getloc()` const
- `streamsize in_avail()`
- `locale pubimbue(const locale &__loc)`
- `int_type sbumpc()`
- `int_type sgetc()`
- `streamsize sgetn(char_type *__s, streamsize __n)`
- `int_type snextc()`
- `int_type sputbackc(char_type __c)`
- `int_type sputc(char_type __c)`
- `streamsize sputn(const char_type *__s, streamsize __n)`
- `int_type sungetc()`
- `basic_streambuf * pubsetbuf(char_type *__s, streamsize __n)`
- `pos_type pubseekoff(off_type __off, ios_base::seekdir __way, ios_base::openmode __mode=ios_base::in|ios_base::out)`
- `pos_type pubseekpos(pos_type __sp, ios_base::openmode __mode=ios_base::in|ios_base::out)`
- `int pubsync()`

Protected Member Functions

- `basic_streambuf()`
- `basic_streambuf(const basic_streambuf &)`
- void `__safe_gbump(streamsize __n)`
- void `__safe_pbump(streamsize __n)`
- void `gbump(int __n)`
- virtual void `imbue(const locale &__loc)`
- `basic_streambuf & operator=(const basic_streambuf &)`
- virtual `int_type overflow(int_type __c=traits_type::eof())`
- virtual `int_type pbackfail(int_type __c=traits_type::eof())`
- void `pbump(int __n)`
- virtual `pos_type seekoff(off_type, ios_base::seekdir, ios_base::openmode=ios_base::in|ios_base::out)`
- virtual `pos_type seekpos(pos_type, ios_base::openmode=ios_base::in|ios_base::out)`
- virtual `basic_streambuf< char_type, _Traits > * setbuf(char_type *, streamsize)`
- void `setg(char_type *__gbeg, char_type *__gnext, char_type *__gend)`
- void `setp(char_type *__pbeg, char_type *__pend)`

- virtual [streamsize showmanyc](#) ()
- void **swap** ([basic_streambuf](#) &__sb)
- virtual int [sync](#) ()
- virtual [int_type uflow](#) ()
- virtual [int_type underflow](#) ()
- virtual [streamsize xsgetn](#) ([char_type](#) *__s, [streamsize](#) __n)
- virtual [streamsize xspn](#) (const [char_type](#) *__s, [streamsize](#) __n)

- [char_type](#) * [eback](#) () const
- [char_type](#) * [gptr](#) () const
- [char_type](#) * [egptr](#) () const

- [char_type](#) * [pbase](#) () const
- [char_type](#) * [pptr](#) () const
- [char_type](#) * [epptr](#) () const

Protected Attributes

- [locale _M_buf_locale](#)
- [char_type](#) * [_M_in_beg](#)
- [char_type](#) * [_M_in_cur](#)
- [char_type](#) * [_M_in_end](#)
- [char_type](#) * [_M_out_beg](#)
- [char_type](#) * [_M_out_cur](#)
- [char_type](#) * [_M_out_end](#)

Friends

- template<bool _IsMove, typename _CharT2 >
[__gnu_cxx::__enable_if](#)< [__is_char](#)< _CharT2 >::__value, _CharT2 * >::__type [__copy_move_a2](#)
([istreambuf_iterator](#)< _CharT2 >, [istreambuf_iterator](#)< _CharT2 >, _CharT2 *)
- [streamsize](#) [__copy_streambufs_eof](#) ([basic_streambuf](#) *, [basic_streambuf](#) *, bool &)
- class **basic_ios**< [char_type](#), [traits_type](#) >
- class **basic_istream**< [char_type](#), [traits_type](#) >
- class **basic_ostream**< [char_type](#), [traits_type](#) >
- template<typename _CharT2 >
[__gnu_cxx::__enable_if](#)< [__is_char](#)< _CharT2 >::__value, [istreambuf_iterator](#)< _CharT2 > >::__type [find](#)
([istreambuf_iterator](#)< _CharT2 >, [istreambuf_iterator](#)< _CharT2 >, const _CharT2 &)
- template<typename _CharT2, typename _Traits2, typename _Alloc >
[basic_istream](#)< _CharT2, _Traits2 > & [getline](#) ([basic_istream](#)< _CharT2, _Traits2 > &, [basic_string](#)< _CharT2,
_Traits2, _Alloc > &, _CharT2)
- class **istreambuf_iterator**< [char_type](#), [traits_type](#) >
- template<typename _CharT2, typename _Traits2 >
[basic_istream](#)< _CharT2, _Traits2 > & [operator](#)>> ([basic_istream](#)< _CharT2, _Traits2 > &, _CharT2 *)
- template<typename _CharT2, typename _Traits2, typename _Alloc >
[basic_istream](#)< _CharT2, _Traits2 > & [operator](#)>> ([basic_istream](#)< _CharT2, _Traits2 > &, [basic_string](#)<
_CharT2, _Traits2, _Alloc > &)
- class **ostreambuf_iterator**< [char_type](#), [traits_type](#) >

5.623.1 Detailed Description

```
template<typename _CharT, typename _Traits>
class std::basic_streambuf<_CharT, _Traits>
```

The actual work of input and output (interface).

Template Parameters

<code>_CharT</code>	Type of character stream.
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .

This is a base class. Derived stream buffers each control a pair of character sequences: one for input, and one for output.

Section [27.5.1] of the standard describes the requirements and behavior of stream buffer classes. That section (three paragraphs) is reproduced here, for simplicity and accuracy.

- Stream buffers can impose various constraints on the sequences they control. Some constraints are:
 - The controlled input sequence can be not readable.
 - The controlled output sequence can be not writable.
 - The controlled sequences can be associated with the contents of other representations for character sequences, such as external files.
 - The controlled sequences can support operations *directly* to or from associated sequences.
 - The controlled sequences can impose limitations on how the program can read characters from a sequence, write characters to a sequence, put characters back into an input sequence, or alter the stream position.
- Each sequence is characterized by three pointers which, if non-null, all point into the same `charT` array object. The array object represents, at any moment, a (sub)sequence of characters from the sequence. Operations performed on a sequence alter the values stored in these pointers, perform reads and writes directly to or from associated sequences, and alter *the stream position* and conversion state as needed to maintain this subsequence relationship. The three pointers are:
 - the *beginning pointer*, or lowest element address in the array (called *xbeg* here);
 - the *next pointer*, or next element address that is a current candidate for reading or writing (called *xnext* here);
 - the *end pointer*, or first element address beyond the end of the array (called *xend* here).
- The following semantic constraints shall always apply for any set of three pointers for a sequence, using the pointer names given immediately above:
 - If *xnext* is not a null pointer, then *xbeg* and *xend* shall also be non-null pointers into the same `charT` array, as described above; otherwise, *xbeg* and *xend* shall also be null.
 - If *xnext* is not a null pointer and *xnext* < *xend* for an output sequence, then a *write position* is available. In this case, **xnext* shall be assignable as the next element to write (to put, or to store a character value, into the sequence).
 - If *xnext* is not a null pointer and *xbeg* < *xnext* for an input sequence, then a *putback position* is available. In this case, *xnext*[-1] shall have a defined value and is the next (preceding) element to store a character that is put back into the input sequence.
 - If *xnext* is not a null pointer and *xnext* < *xend* for an input sequence, then a *read position* is available. In this case, **xnext* shall have a defined value and is the next element to read (to get, or to obtain a character value, from the sequence).

Definition at line 80 of file `iosfwd`.

5.623.2 Member Typedef Documentation

5.623.2.1 `template<typename _CharT, typename _Traits> typedef basic_streambuf<char_type, traits_type>
std::basic_streambuf< _CharT, _Traits >::__streambuf_type`

This is a non-standard type.

Definition at line 138 of file streambuf.

5.623.2.2 `template<typename _CharT, typename _Traits> typedef _CharT std::basic_streambuf< _CharT, _Traits
>::char_type`

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

Definition at line 129 of file streambuf.

5.623.2.3 `template<typename _CharT, typename _Traits> typedef traits_type::int_type std::basic_streambuf< _CharT, _Traits
>::int_type`

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

Definition at line 131 of file streambuf.

5.623.2.4 `template<typename _CharT, typename _Traits> typedef traits_type::off_type std::basic_streambuf< _CharT, _Traits
>::off_type`

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

Definition at line 133 of file streambuf.

5.623.2.5 `template<typename _CharT, typename _Traits> typedef traits_type::pos_type std::basic_streambuf< _CharT, _Traits
>::pos_type`

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

Definition at line 132 of file streambuf.

5.623.2.6 `template<typename _CharT, typename _Traits> typedef _Traits std::basic_streambuf< _CharT, _Traits
>::traits_type`

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

Definition at line 130 of file streambuf.

5.623.3 Constructor & Destructor Documentation

5.623.3.1 `template<typename _CharT, typename _Traits> virtual std::basic_streambuf<_CharT, _Traits>::~~basic_streambuf() [inline], [virtual]`

Destructor deallocates no buffer space.

Definition at line 197 of file streambuf.

5.623.3.2 `template<typename _CharT, typename _Traits> std::basic_streambuf<_CharT, _Traits>::basic_streambuf() [inline], [protected]`

Base constructor.

Only called from derived constructors, and sets up all the buffer data to zero, including the pointers described in the basic_streambuf class description. Note that, as a result,

- the class starts with no read nor write positions available,
- this is not an error

Definition at line 463 of file streambuf.

5.623.4 Member Function Documentation

5.623.4.1 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::eback() const [inline], [protected]`

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- eback() returns the beginning pointer for the input sequence
- gptr() returns the next pointer for the input sequence
- egptr() returns the end pointer for the input sequence

Definition at line 482 of file streambuf.

Referenced by std::basic_filebuf<_CharT, _Traits>::imbue(), std::basic_filebuf<_CharT, _Traits>::pbackfail(), std::basic_filebuf<_CharT, _Traits>::seekpos(), std::basic_filebuf<_CharT, _Traits>::underflow(), and std::basic_filebuf<_CharT, _Traits>::xsgetn().

5.623.4.2 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::egptr ()`
`const [inline], [protected]`

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Definition at line 488 of file `streambuf`.

Referenced by `std::ostreambuf_iterator< _CharT, _Traits >::failed()`, `std::basic_filebuf< _CharT, _Traits >::seekpos()`, `std::basic_filebuf< _CharT, _Traits >::showmanyc()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

5.623.4.3 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::epptr ()`
`const [inline], [protected]`

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

Definition at line 535 of file `streambuf`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::xspn()`.

5.623.4.4 `template<typename _CharT, typename _Traits> void std::basic_streambuf< _CharT, _Traits >::gbump (int __n)`
`[inline], [protected]`

Moving the read position.

Parameters

<code>__n</code>	The delta by which to move.
------------------	-----------------------------

This just advances the read position without returning any data.

Definition at line 498 of file `streambuf`.

Referenced by std::basic_filebuf< _CharT, _Traits >::pbackfail(), and std::basic_filebuf< _CharT, _Traits >::xsgetn().

5.623.4.5 `template<typename _CharT, typename _Traits> locale std::basic_streambuf< _CharT, _Traits >::getloc () const`
`[inline]`

Locale access.

Returns

The current locale in effect.

If pubimbue(loc) has been called, then the most recent `loc` is returned. Otherwise the global locale in effect at the time of construction is returned.

Definition at line 226 of file streambuf.

5.623.4.6 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::gptr ()`
`const [inline], [protected]`

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Definition at line 485 of file streambuf.

Referenced by std::ostreambuf_iterator< _CharT, _Traits >::failed(), std::basic_filebuf< _CharT, _Traits >::imbue(), std::basic_filebuf< _CharT, _Traits >::pbackfail(), std::basic_filebuf< _CharT, _Traits >::seekpos(), std::basic_filebuf< _CharT, _Traits >::showmanyc(), std::basic_filebuf< _CharT, _Traits >::underflow(), and std::basic_filebuf< _CharT, _Traits >::xsgetn().

5.623.4.7 `template<typename _CharT, typename _Traits> virtual void std::basic_streambuf< _CharT, _Traits >::imbue (const`
`locale & __loc) [inline], [protected], [virtual]`

Changes translations.

Parameters

<code>__loc</code>	A new locale.
--------------------	---------------

Translations done during I/O which depend on the current locale are changed by this call. The standard adds, *Between invocations of this function a class derived from streambuf can safely cache results of calls to locale functions and to members of facets so obtained.*

Note

Base class version does nothing.

Reimplemented in [std::basic_filebuf<_CharT, _Traits>](#), [std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>](#), and [std::basic_filebuf<char_type, traits_type>](#).

Definition at line 576 of file streambuf.

5.623.4.8 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf<_CharT, _Traits>::in_avail ()`
`[inline]`

Looking ahead into the stream.

Returns

The number of characters available.

If a read position is available, returns the number of characters available for reading before the buffer must be refilled. Otherwise returns the derived `showmanyc()`.

Definition at line 284 of file streambuf.

5.623.4.9 `template<typename _CharT, typename _Traits> virtual int_type std::basic_streambuf<_CharT, _Traits>::overflow`
`(int_type __c=traits_type::eof()) [inline], [protected], [virtual]`

Consumes data from the buffer; writes to the controlled sequence.

Parameters

<code>__c</code>	An additional character to consume.
------------------	-------------------------------------

Returns

`eof()` to indicate failure, something else (usually `__c`, or `not_eof()`)

Informally, this function is called when the output buffer is full (or does not exist, as buffering need not actually be done). If a buffer exists, it is *consumed*, with *some effect* on the controlled sequence. (Typically, the buffer is written out to the sequence verbatim.) In either case, the character `c` is also written out, if `__c` is not `eof()`.

For a formal definition of this function, see a good text such as Langer & Kreft, or [27.5.2.4.5]/3-7.

A functioning output streambuf can be created by overriding only this function (no buffer area will be used).

Note

Base class version does nothing, returns `eof()`.

Reimplemented in [std::wbuffer_convert<_Codecvt, _Elem, _Tr>](#), [std::basic_filebuf<_CharT, _Traits>](#), [std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>](#), [std::basic_filebuf<char_type, traits_type>](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>](#), and [__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>](#).

Definition at line 768 of file streambuf.

```
5.623.4.10 template<typename _CharT, typename _Traits> virtual int_type std::basic_streambuf<_CharT, _Traits>::pbackfail( int_type __c=traits_type::eof() ) [inline], [protected], [virtual]
```

Tries to back up the input sequence.

Parameters

<code>__c</code>	The character to be inserted back into the sequence.
------------------	--

Returns

eof() on failure, *some other value* on success

Postcondition

The constraints of gptr(), eback(), and pptr() are the same as for underflow().

Note

Base class version does nothing, returns eof().

Reimplemented in [std::basic_filebuf<_CharT, _Traits>](#), [std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>](#), [std::basic_filebuf<char_type, traits_type>](#), [std::basic_stringbuf<_CharT, _Traits, _Alloc>](#), and [__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>](#).

Definition at line 724 of file streambuf.

```
5.623.4.11 template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::pbase( ) const [inline], [protected]
```

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- pbase() returns the beginning pointer for the output sequence
- pptr() returns the next pointer for the output sequence
- epptr() returns the end pointer for the output sequence

Definition at line 529 of file streambuf.

Referenced by [std::basic_filebuf<_CharT, _Traits>::overflow\(\)](#), [std::basic_filebuf<_CharT, _Traits>::seekoff\(\)](#), [std::basic_filebuf<_CharT, _Traits>::seekpos\(\)](#), [std::basic_filebuf<_CharT, _Traits>::sync\(\)](#), and [std::basic_filebuf<_CharT, _Traits>::xspn\(\)](#).

```
5.623.4.12 template<typename _CharT, typename _Traits> void std::basic_streambuf<_CharT, _Traits>::pbump( int __n ) [inline], [protected]
```

Moving the write position.

Parameters

<code>_↔</code>	The delta by which to move.
<code>_n</code>	

This just advances the write position without returning any data.

Definition at line 545 of file streambuf.

Referenced by `std::basic_filebuf<_CharT, _Traits>::overflow()`.

5.623.4.13 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::pptr ()`
`const [inline], [protected]`

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

Definition at line 532 of file streambuf.

Referenced by `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_filebuf<_CharT, _Traits>::seekoff()`, `std::basic_filebuf<_CharT, _Traits>::seekpos()`, `std::basic_filebuf<_CharT, _Traits>::sync()`, and `std::basic_filebuf<_CharT, _Traits>::xsputn()`.

5.623.4.14 `template<typename _CharT, typename _Traits> locale std::basic_streambuf<_CharT, _Traits>::pubimbue (`
`const locale &__loc) [inline]`

Entry point for `imbue()`.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls the derived `imbue(__loc)`.

Definition at line 209 of file streambuf.

```
5.623.4.15 template<typename _CharT, typename _Traits> pos_type std::basic_streambuf<_CharT, _Traits>::pubseekoff (
    off_type __off, ios_base::seekdir __way, ios_base::openmode __mode = ios_base::in | ios_base::out )
    [inline]
```

Alters the stream position.

Parameters

<code>__off</code>	Offset.
<code>__way</code>	Value for <code>ios_base::seekdir</code> .
<code>__mode</code>	Value for <code>ios_base::openmode</code> .

Calls virtual seekoff function.

Definition at line 251 of file streambuf.

```
5.623.4.16 template<typename _CharT, typename _Traits> pos_type std::basic_streambuf<_CharT, _Traits>::pubseekpos
    ( pos_type __sp, ios_base::openmode __mode = ios_base::in | ios_base::out ) [inline]
```

Alters the stream position.

Parameters

<code>__sp</code>	Position
<code>__mode</code>	Value for <code>ios_base::openmode</code> .

Calls virtual seekpos function.

Definition at line 263 of file streambuf.

```
5.623.4.17 template<typename _CharT, typename _Traits> basic_streambuf* std::basic_streambuf<_CharT, _Traits>
    >::pubsetbuf ( char_type * __s, streamsize __n ) [inline]
```

Entry points for derived buffer functions.

The public versions of `pubfoo` dispatch to the protected derived `foo` member functions, passing the arguments (if any) and returning the result unchanged.

Definition at line 239 of file streambuf.

```
5.623.4.18 template<typename _CharT, typename _Traits> int std::basic_streambuf<_CharT, _Traits>::pubsync ( )
    [inline]
```

Calls virtual sync function.

Definition at line 271 of file streambuf.

Referenced by `std::basic_istream<_CharT, _Traits>::sync()`.

5.623.4.19 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf< _CharT, _Traits >::sbumpc ()`
`[inline]`

Getting the next character.

Returns

The next character, or eof.

If the input read position is available, returns that character and increments the read pointer, otherwise calls and returns `uflow()`.

Definition at line 316 of file `streambuf`.

Referenced by `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, and `std::istreambuf_iterator< _CharT, _Traits >::operator++()`.

5.623.4.20 `template<typename _CharT, typename _Traits> virtual pos_type std::basic_streambuf< _CharT, _Traits`
`>::seekoff (off_type , ios_base::seekdir , ios_base::openmode = ios_base::in | ios_base::out)`
`[inline], [protected], [virtual]`

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented in `std::basic_filebuf< _CharT, _Traits >`, `std::basic_filebuf< _CharT, encoding_char_traits< _CharT >`
`>`, `std::basic_filebuf< char_type, traits_type >`, and `std::basic_stringbuf< _CharT, _Traits, _Alloc >`.

Definition at line 602 of file `streambuf`.

5.623.4.21 `template<typename _CharT, typename _Traits> virtual pos_type std::basic_streambuf< _CharT, _Traits`
`>::seekpos (pos_type , ios_base::openmode = ios_base::in | ios_base::out) [inline],`
`[protected], [virtual]`

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented in `std::basic_filebuf< _CharT, _Traits >`, `std::basic_filebuf< _CharT, encoding_char_traits< _CharT >`
`>`, `std::basic_filebuf< char_type, traits_type >`, and `std::basic_stringbuf< _CharT, _Traits, _Alloc >`.

Definition at line 614 of file `streambuf`.

```
5.623.4.22 template<typename _CharT, typename _Traits> virtual basic_streambuf<char_type, _Traits>*
std::basic_streambuf<_CharT, _Traits>::setbuf ( char_type *, streamsize ) [inline],
[protected], [virtual]
```

Manipulates the buffer.

Each derived class provides its own appropriate behavior. See the next-to-last paragraph of <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html#io.streambuf.buffering> for more on this function.

Note

Base class version does nothing, returns `this`.

Reimplemented in `std::basic_filebuf<_CharT, _Traits>`, `std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>`, `std::basic_filebuf<char_type, traits_type>`, and `std::basic_stringbuf<_CharT, _Traits, _Alloc>`.

Definition at line 591 of file `streambuf`.

```
5.623.4.23 template<typename _CharT, typename _Traits> void std::basic_streambuf<_CharT, _Traits>::setg ( char_type
* __gbeg, char_type * __gnext, char_type * __gend ) [inline], [protected]
```

Setting the three read area pointers.

Parameters

<code>__gbeg</code>	A pointer.
<code>__gnext</code>	A pointer.
<code>__gend</code>	A pointer.

Postcondition

`__gbeg == eback()`, `__gnext == gptr()`, and `__gend == egptr()`

Definition at line 509 of file `streambuf`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::xsgetn()`.

```
5.623.4.24 template<typename _CharT, typename _Traits> void std::basic_streambuf<_CharT, _Traits>::setp ( char_type
* __pbeg, char_type * __pend ) [inline], [protected]
```

Setting the three write area pointers.

Parameters

<code>__pbeg</code>	A pointer.
<code>__pend</code>	A pointer.

Postcondition

`__pbeg == pbase()`, `__pbeg == pptr()`, and `__pend == epptr()`

Definition at line 555 of file `streambuf`.

5.623.4.25 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::sgetc ()`
`[inline]`

Getting the next character.

Returns

The next character, or `eof`.

If the input read position is available, returns that character, otherwise calls and returns `underflow()`. Does not move the read position after fetching the character.

Definition at line 338 of file `streambuf`.

Referenced by `std::istreambuf_iterator<_CharT, _Traits>::equal()`, `std::ostreambuf_iterator<_CharT, _Traits>::failed()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, and `std::basic_streambuf<_CharT, _Traits>::xsputn()`.

5.623.4.26 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf<_CharT, _Traits>::sgetn (`
`char_type *__s, streamsize __n) [inline]`

Entry point for `xsgetn`.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	A count.

Returns `xsgetn(__s, __n)`. The effect is to fill `__s[0]` through `__s[__n-1]` with characters from the input sequence, if possible.

Definition at line 357 of file `streambuf`.

5.623.4.27 `template<typename _CharT, typename _Traits> virtual streamsize std::basic_streambuf<_CharT, _Traits>::showmanyc () [inline], [protected], [virtual]`

Investigating the data available.

Returns

An estimate of the number of characters available in the input sequence, or -1.

If it returns a positive value, then successive calls to `underflow()` will not return `traits::eof()` until at least that number of characters have been supplied. If `showmanyc()` returns -1, then calls to `underflow()` or `uflow()` will fail. [27.5.2.4.3]/1

Note

Base class version does nothing, returns zero.

The standard adds that *the intention is not only that the calls [to `underflow` or `uflow`] will not return `eof()` but that they will return immediately.*

The standard adds that *the morphemes of `showmanyc` are **es-how-many-see**, not **show-manic**.*

Reimplemented in [std::basic_filebuf< _CharT, _Traits >](#), [std::basic_filebuf< _CharT, encoding_char_traits< _CharT >](#), [std::basic_filebuf< char_type, traits_type >](#), and [std::basic_stringbuf< _CharT, _Traits, _Alloc >](#).

Definition at line 649 of file `streambuf`.

```
5.623.4.28 template<typename _CharT, typename _Traits> int_type std::basic_streambuf< _CharT, _Traits >::snextc ( )
        [inline]
```

Getting the next character.

Returns

The next character, or eof.

Calls `sbumpc()`, and if that function returns `traits::eof()`, so does this function. Otherwise, `sgetc()`.

Definition at line 298 of file `streambuf`.

Referenced by `std::ostreambuf_iterator< _CharT, _Traits >::failed()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_streambuf< _CharT, _Traits >::xsputn()`.

```
5.623.4.29 template<typename _CharT, typename _Traits> int_type std::basic_streambuf< _CharT, _Traits >::sputbackc (
        char_type _c ) [inline]
```

Pushing characters back into the input stream.

Parameters

<code>_↵</code>	The character to push back.
<code>_c</code>	

Returns

The previous character, if possible.

Similar to `sungetc()`, but `__c` is pushed onto the stream instead of *the previous character*. If successful, the next character fetched from the input stream will be `__c`.

Definition at line 372 of file `streambuf`.

Referenced by `std::basic_istream<_CharT, _Traits>::putback()`.

5.623.4.30 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::sputc (char_type __c) [inline]`

Entry point for all single-character output functions.

Parameters

<code>__c</code>	A character to output.
------------------	------------------------

Returns

`__c`, if possible.

One of two public output functions.

If a write position is available for the output sequence (i.e., the buffer is not full), stores `__c` in that position, increments the position, and returns `traits::to_int_type(__c)`. If a write position is not available, returns `overflow(__c)`.

Definition at line 424 of file `streambuf`.

Referenced by `std::basic_istream<_CharT, _Traits>::get()`, `std::ostreambuf_iterator<_CharT, _Traits>::operator=()`, and `std::basic_streambuf<_CharT, _Traits>::xsputn()`.

5.623.4.31 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf<_CharT, _Traits>::sputn (const char_type * __s, streamsize __n) [inline]`

Entry point for all single-character output functions.

Parameters

<code>__s</code>	A buffer read area.
<code>__n</code>	A count.

One of two public output functions.

Returns `xspn(__s,__n)`. The effect is to write `__s[0]` through `__s[__n-1]` to the output sequence, if possible.

Definition at line 450 of file `streambuf`.

Referenced by `std::ostreambuf_iterator< _CharT, _Traits >::failed()`.

5.623.4.32 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf< _CharT, _Traits >::sungetc ()`
[inline]

Moving backwards in the input stream.

Returns

The previous character, if possible.

If a putback position is available, this function decrements the input pointer and returns that character. Otherwise, calls and returns `pbackfail()`. The effect is to *unget* the last character *gotten*.

Definition at line 397 of file `streambuf`.

Referenced by `std::basic_istream< _CharT, _Traits >::unget()`.

5.623.4.33 `template<typename _CharT, typename _Traits> virtual int std::basic_streambuf< _CharT, _Traits >::sync (void)`
[inline], [protected], [virtual]

Synchronizes the buffer arrays with the controlled sequences.

Returns

-1 on failure.

Each derived class provides its own appropriate behavior, including the definition of *failure*.

Note

Base class version does nothing, returns zero.

Reimplemented in `std::basic_filebuf< _CharT, _Traits >`, `std::basic_filebuf< _CharT, encoding_char_traits< _CharT > >`, `std::basic_filebuf< char_type, traits_type >`, `std::wbuffer_convert< _Codecvt, _Elem, _Tr >`, and `__gnu_cxx::__stdio_sync_filebuf< _CharT, _Traits >`.

Definition at line 627 of file `streambuf`.

5.623.4.34 `template<typename _CharT, typename _Traits> virtual int_type std::basic_streambuf< _CharT, _Traits >::uflow () [inline], [protected], [virtual]`

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function does the same thing as `underflow()`, and in fact is required to call that function. It also returns the new character, like `underflow()` does. However, this function also moves the read position forward by one.

Reimplemented in [__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >](#).

Definition at line 700 of file `streambuf`.

5.623.4.35 `template<typename _CharT, typename _Traits> virtual int_type std::basic_streambuf< _CharT, _Traits >::underflow () [inline], [protected], [virtual]`

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function is called when the input buffer is exhausted (or does not exist, as buffering need not actually be done). If a buffer exists, it is *refilled*. In either case, the next available character is returned, or `traits::eof()` to indicate a null pending sequence.

For a formal definition of the pending sequence, see a good text such as Langer & Kreft, or [27.5.2.4.3]/7-14.

A functioning input streambuf can be created by overriding only this function (no buffer area will be used). For an example, see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html>

Note

Base class version does nothing, returns `eof()`.

Reimplemented in [std::wbuffer_convert< _Codecvt, _Elem, _Tr >](#), [std::basic_filebuf< _CharT, _Traits >](#), [std::basic_filebuf< _CharT, encoding_char_traits< _CharT > >](#), [std::basic_filebuf< char_type, traits_type >](#), [std::basic_stringbuf< _CharT, _Traits, _Alloc >](#), and [__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >](#).

Definition at line 687 of file `streambuf`.

Referenced by `std::ostreambuf_iterator< _CharT, _Traits >::failed()`.

5.623.4.36 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf< _CharT, _Traits >::xsgetn (char_type * __s, streamsize __n) [protected], [virtual]`

Multiple character extraction.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to assign.

Returns

The number of characters assigned.

Fills `__s[0]` through `__s[__n-1]` with characters from the input sequence, as if by `sbumpc()`. Stops when either `__n` characters have been copied, or when `traits::eof()` would be copied.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented in [std::basic_filebuf<_CharT, _Traits>](#), [std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>](#), and [std::basic_filebuf<char_type, traits_type>](#).

Definition at line 46 of file `streambuf.tcc`.

References `std::min()`, and `std::basic_streambuf<_CharT, _Traits>::xspn()`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::xsgetn()`.

5.623.4.37 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf<_CharT, _Traits>::xspn (const char_type * __s, streamsize __n) [protected], [virtual]`

Multiple character insertion.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to write.

Returns

The number of characters written.

Writes `__s[0]` through `__s[__n-1]` to the output sequence, as if by `sputc()`. Stops when either `n` characters have been copied, or when `sputc()` would return `traits::eof()`.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented in [std::basic_filebuf<_CharT, _Traits>](#), [std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>](#), and [std::basic_filebuf<char_type, traits_type>](#).

Definition at line 80 of file streambuf.tcc.

References `std::min()`, `std::basic_streambuf<_CharT, _Traits>::sgetc()`, `std::basic_streambuf<_CharT, _Traits>::snextc()`, and `std::basic_streambuf<_CharT, _Traits>::sputc()`.

Referenced by `std::basic_streambuf<_CharT, _Traits>::xsgetn()`, and `std::basic_filebuf<_CharT, _Traits>::xsputn()`.

5.623.5 Member Data Documentation

5.623.5.1 `template<typename _CharT, typename _Traits> locale std::basic_streambuf<_CharT, _Traits>::_M_buf_locale`
[protected]

Current locale setting.

Definition at line 192 of file streambuf.

Referenced by `std::basic_filebuf<_CharT, _Traits>::basic_filebuf()`.

5.623.5.2 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::_M_in_beg`
[protected]

Start of get area.

Definition at line 184 of file streambuf.

5.623.5.3 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::_M_in_cur`
[protected]

Current read area.

Definition at line 185 of file streambuf.

5.623.5.4 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::_M_in_end`
[protected]

End of get area.

Definition at line 186 of file streambuf.

5.623.5.5 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::_M_out_beg`
[protected]

Start of put area.

Definition at line 187 of file streambuf.

5.623.5.6 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::_M_out_cur`
`[protected]`

Current put area.

Definition at line 188 of file `streambuf`.

5.623.5.7 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf<_CharT, _Traits>::_M_out_end`
`[protected]`

End of put area.

Definition at line 189 of file `streambuf`.

The documentation for this class was generated from the following files:

- [iosfwd](#)
- [streambuf](#)
- [streambuf.tcc](#)

5.624 `std::basic_string<_CharT, _Traits, _Alloc>` Class Template Reference

Public Types

- `typedef _Alloc allocator_type`
- `typedef __gnu_cxx::__normal_iterator< const_pointer, basic_string > const_iterator`
- `typedef _CharT_alloc_type::const_pointer const_pointer`
- `typedef _CharT_alloc_type::const_reference const_reference`
- `typedef std::reverse_iterator< const_iterator > const_reverse_iterator`
- `typedef _CharT_alloc_type::difference_type difference_type`
- `typedef __gnu_cxx::__normal_iterator< pointer, basic_string > iterator`
- `typedef _CharT_alloc_type::pointer pointer`
- `typedef _CharT_alloc_type::reference reference`
- `typedef std::reverse_iterator< iterator > reverse_iterator`
- `typedef _CharT_alloc_type::size_type size_type`
- `typedef _Traits traits_type`
- `typedef _Traits::char_type value_type`

Public Member Functions

- [basic_string](#) ()
- [basic_string](#) (const [_Alloc](#) &__a)
- [basic_string](#) (const [basic_string](#) &__str)
- [basic_string](#) (const [basic_string](#) &__str, size_type __pos, size_type __n=[npos](#))
- [basic_string](#) (const [basic_string](#) &__str, size_type __pos, size_type __n, const [_Alloc](#) &__a)
- [basic_string](#) (const [_CharT](#) *__s, size_type __n, const [_Alloc](#) &__a=[_Alloc](#)())
- [basic_string](#) (const [_CharT](#) *__s, const [_Alloc](#) &__a=[_Alloc](#)())
- [basic_string](#) (size_type __n, [_CharT](#) __c, const [_Alloc](#) &__a=[_Alloc](#)())
- [basic_string](#) ([basic_string](#) &&__str) noexcept
- [basic_string](#) ([initializer_list](#)< [_CharT](#) > __l, const [_Alloc](#) &__a=[_Alloc](#)())
- template<class [_InputIterator](#) >
 [basic_string](#) ([_InputIterator](#) __beg, [_InputIterator](#) __end, const [_Alloc](#) &__a=[_Alloc](#)())
- ~[basic_string](#) () noexcept
- template<typename [_InputIterator](#) >
 [basic_string](#)< [_CharT](#), [_Traits](#), [_Alloc](#) > & [_M_replace_dispatch](#) (iterator __i1, iterator __i2, [_InputIterator](#) __k1, [_InputIterator](#) __k2, __false_type)
- template<typename [_InIterator](#) >
 [_CharT](#) * [_S_construct](#) ([_InIterator](#) __beg, [_InIterator](#) __end, const [_Alloc](#) &__a, [forward_iterator_tag](#))
- [basic_string](#) & [append](#) (const [basic_string](#) &__str)
- [basic_string](#) & [append](#) (const [basic_string](#) &__str, size_type __pos, size_type __n)
- [basic_string](#) & [append](#) (const [_CharT](#) *__s, size_type __n)
- [basic_string](#) & [append](#) (const [_CharT](#) *__s)
- [basic_string](#) & [append](#) (size_type __n, [_CharT](#) __c)
- [basic_string](#) & [append](#) ([initializer_list](#)< [_CharT](#) > __l)
- template<class [_InputIterator](#) >
 [basic_string](#) & [append](#) ([_InputIterator](#) __first, [_InputIterator](#) __last)
- [basic_string](#) & [assign](#) (const [basic_string](#) &__str)
- [basic_string](#) & [assign](#) ([basic_string](#) &&__str)
- [basic_string](#) & [assign](#) (const [basic_string](#) &__str, size_type __pos, size_type __n)
- [basic_string](#) & [assign](#) (const [_CharT](#) *__s, size_type __n)
- [basic_string](#) & [assign](#) (const [_CharT](#) *__s)
- [basic_string](#) & [assign](#) (size_type __n, [_CharT](#) __c)
- template<class [_InputIterator](#) >
 [basic_string](#) & [assign](#) ([_InputIterator](#) __first, [_InputIterator](#) __last)
- [basic_string](#) & [assign](#) ([initializer_list](#)< [_CharT](#) > __l)
- const_reference [at](#) (size_type __n) const
- reference [at](#) (size_type __n)
- reference [back](#) ()
- const_reference [back](#) () const noexcept
- iterator [begin](#) ()
- const_iterator [begin](#) () const noexcept
- const [_CharT](#) * [c_str](#) () const noexcept
- size_type [capacity](#) () const noexcept
- const_iterator [cbegin](#) () const noexcept
- const_iterator [cend](#) () const noexcept
- void [clear](#) ()
- int [compare](#) (const [basic_string](#) &__str) const
- int [compare](#) (size_type __pos, size_type __n, const [basic_string](#) &__str) const

- int [compare](#) (size_type __pos1, size_type __n1, const [basic_string](#) &__str, size_type __pos2, size_type __n2) const
- int [compare](#) (const _CharT *__s) const
- int [compare](#) (size_type __pos, size_type __n1, const _CharT *__s) const
- int [compare](#) (size_type __pos, size_type __n1, const _CharT *__s, size_type __n2) const
- size_type [copy](#) (_CharT *__s, size_type __n, size_type __pos=0) const
- [const_reverse_iterator](#) [cbegin](#) () const noexcept
- [const_reverse_iterator](#) [crend](#) () const noexcept
- const _CharT * [data](#) () const noexcept
- bool [empty](#) () const noexcept
- iterator [end](#) ()
- const_iterator [end](#) () const noexcept
- [basic_string](#) & [erase](#) (size_type __pos=0, size_type __n=[npos](#))
- iterator [erase](#) (iterator __position)
- iterator [erase](#) (iterator __first, iterator __last)
- size_type [find](#) (const _CharT *__s, size_type __pos, size_type __n) const
- size_type [find](#) (const [basic_string](#) &__str, size_type __pos=0) const noexcept
- size_type [find](#) (const _CharT *__s, size_type __pos=0) const
- size_type [find](#) (_CharT __c, size_type __pos=0) const noexcept
- size_type [find_first_not_of](#) (const [basic_string](#) &__str, size_type __pos=0) const noexcept
- size_type [find_first_not_of](#) (const _CharT *__s, size_type __pos, size_type __n) const
- size_type [find_first_not_of](#) (const _CharT *__s, size_type __pos=0) const
- size_type [find_first_not_of](#) (_CharT __c, size_type __pos=0) const noexcept
- size_type [find_first_of](#) (const [basic_string](#) &__str, size_type __pos=0) const noexcept
- size_type [find_first_of](#) (const _CharT *__s, size_type __pos, size_type __n) const
- size_type [find_first_of](#) (const _CharT *__s, size_type __pos=0) const
- size_type [find_first_of](#) (_CharT __c, size_type __pos=0) const noexcept
- size_type [find_last_not_of](#) (const [basic_string](#) &__str, size_type __pos=[npos](#)) const noexcept
- size_type [find_last_not_of](#) (const _CharT *__s, size_type __pos, size_type __n) const
- size_type [find_last_not_of](#) (const _CharT *__s, size_type __pos=[npos](#)) const
- size_type [find_last_not_of](#) (_CharT __c, size_type __pos=[npos](#)) const noexcept
- size_type [find_last_of](#) (const [basic_string](#) &__str, size_type __pos=[npos](#)) const noexcept
- size_type [find_last_of](#) (const _CharT *__s, size_type __pos, size_type __n) const
- size_type [find_last_of](#) (const _CharT *__s, size_type __pos=[npos](#)) const
- size_type [find_last_of](#) (_CharT __c, size_type __pos=[npos](#)) const noexcept
- reference [front](#) ()
- const_reference [front](#) () const noexcept
- allocator_type [get_allocator](#) () const noexcept
- void [insert](#) (iterator __p, size_type __n, _CharT __c)
- template<class _InputIterator>
 - void [insert](#) (iterator __p, _InputIterator __beg, _InputIterator __end)
- void [insert](#) (iterator __p, [initializer_list](#)<_CharT> __l)
- [basic_string](#) & [insert](#) (size_type __pos1, const [basic_string](#) &__str)
- [basic_string](#) & [insert](#) (size_type __pos1, const [basic_string](#) &__str, size_type __pos2, size_type __n)
- [basic_string](#) & [insert](#) (size_type __pos, const _CharT *__s, size_type __n)
- [basic_string](#) & [insert](#) (size_type __pos, const _CharT *__s)
- [basic_string](#) & [insert](#) (size_type __pos, size_type __n, _CharT __c)
- iterator [insert](#) (iterator __p, _CharT __c)
- size_type [length](#) () const noexcept
- size_type [max_size](#) () const noexcept
- [basic_string](#) & [operator+=](#) (const [basic_string](#) &__str)

- [basic_string](#) & [operator+=](#) (const [_CharT](#) *__s)
- [basic_string](#) & [operator+=](#) ([_CharT](#) __c)
- [basic_string](#) & [operator+=](#) ([initializer_list](#)< [_CharT](#) > __l)
- [basic_string](#) & [operator=](#) (const [basic_string](#) &__str)
- [basic_string](#) & [operator=](#) (const [_CharT](#) *__s)
- [basic_string](#) & [operator=](#) ([_CharT](#) __c)
- [basic_string](#) & [operator=](#) ([basic_string](#) &&__str)
- [basic_string](#) & [operator=](#) ([initializer_list](#)< [_CharT](#) > __l)
- const_reference [operator\[\]](#) (size_type __pos) const noexcept
- reference [operator\[\]](#) (size_type __pos)
- void [pop_back](#) ()
- void [push_back](#) ([_CharT](#) __c)
- [reverse_iterator](#) rbegin ()
- const_reverse_iterator rbegin () const noexcept
- [reverse_iterator](#) rend ()
- const_reverse_iterator rend () const noexcept
- [basic_string](#) & [replace](#) (size_type __pos, size_type __n, const [basic_string](#) &__str)
- [basic_string](#) & [replace](#) (size_type __pos1, size_type __n1, const [basic_string](#) &__str, size_type __pos2, size_type __n2)
- [basic_string](#) & [replace](#) (size_type __pos, size_type __n1, const [_CharT](#) *__s, size_type __n2)
- [basic_string](#) & [replace](#) (size_type __pos, size_type __n1, const [_CharT](#) *__s)
- [basic_string](#) & [replace](#) (size_type __pos, size_type __n1, size_type __n2, [_CharT](#) __c)
- [basic_string](#) & [replace](#) (iterator __i1, iterator __i2, const [basic_string](#) &__str)
- [basic_string](#) & [replace](#) (iterator __i1, iterator __i2, const [_CharT](#) *__s, size_type __n)
- [basic_string](#) & [replace](#) (iterator __i1, iterator __i2, const [_CharT](#) *__s)
- [basic_string](#) & [replace](#) (iterator __i1, iterator __i2, size_type __n, [_CharT](#) __c)
- template<class [_InputIterator](#) >
[basic_string](#) & [replace](#) (iterator __i1, iterator __i2, [_InputIterator](#) __k1, [_InputIterator](#) __k2)
- [basic_string](#) & [replace](#) (iterator __i1, iterator __i2, [_CharT](#) *__k1, [_CharT](#) *__k2)
- [basic_string](#) & [replace](#) (iterator __i1, iterator __i2, const [_CharT](#) *__k1, const [_CharT](#) *__k2)
- [basic_string](#) & [replace](#) (iterator __i1, iterator __i2, iterator __k1, iterator __k2)
- [basic_string](#) & [replace](#) (iterator __i1, iterator __i2, const_iterator __k1, const_iterator __k2)
- [basic_string](#) & [replace](#) (iterator __i1, iterator __i2, [initializer_list](#)< [_CharT](#) > __l)
- void [reserve](#) (size_type __res_arg=0)
- void [resize](#) (size_type __n, [_CharT](#) __c)
- void [resize](#) (size_type __n)
- size_type [rfind](#) (const [basic_string](#) &__str, size_type __pos=[npos](#)) const noexcept
- size_type [rfind](#) (const [_CharT](#) *__s, size_type __pos, size_type __n) const
- size_type [rfind](#) (const [_CharT](#) *__s, size_type __pos=[npos](#)) const
- size_type [rfind](#) ([_CharT](#) __c, size_type __pos=[npos](#)) const noexcept
- void [shrink_to_fit](#) () noexcept
- size_type [size](#) () const noexcept
- [basic_string](#) substr (size_type __pos=0, size_type __n=[npos](#)) const
- void [swap](#) ([basic_string](#) &__s)

Static Public Attributes

- static const size_type [npos](#)

5.624.1 Detailed Description

```
template<typename _CharT, typename _Traits, typename _Alloc>
class std::basic_string<_CharT, _Traits, _Alloc>
```

Managing sequences of characters and character-like objects.

Template Parameters

<code>_CharT</code>	Type of character
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_CharT></code> .

Meets the requirements of a [container](#), a [reversible container](#), and a [sequence](#). Of the [optional sequence requirements](#), only `push_back`, `at`, and array access are supported.

Todo Needs documentation! See http://gcc.gnu.org/onlinedocs/libstdc++/manual/documentation_style.html

Documentation? What's that? Nathan Myers ncm@cantrip.org.

A string looks like this:

```

[basic_string<char_type>]      [_Rep]
_M_dataplus                  _M_length
_M_p ----->                _M_capacity
                               _M_refcount
                               unnamed array of char_type
```

Where the `_M_p` points to the first character in the string, and you cast it to a pointer-to-`_Rep` and subtract 1 to get a pointer to the header.

This approach has the enormous advantage that a string object requires only one allocation. All the ugliness is confined within a single pair of inline functions, which each compile to a single *add* instruction: `_Rep::_M_data()`, and `string↵::_M_rep()`; and the allocation function which gets a block of raw bytes and with room enough and constructs a `_Rep` object at the front.

The reason you want `_M_data` pointing to the character array and not the `_Rep` is so that the debugger can see the string contents. (Probably we should add a non-inline member to get the `_Rep` for the debugger to use, so users can check the actual string length.)

Note that the `_Rep` object is a POD so that you can have a static *empty string* `_Rep` object already *constructed* before static constructors have run. The reference-count encoding is chosen so that a 0 indicates one reference, so you never try to destroy the empty-string `_Rep` object.

All but the last paragraph is considered pretty conventional for a C++ string implementation.

Definition at line 2599 of file `basic_string.h`.

5.624.2 Constructor & Destructor Documentation

5.624.2.1 `template<typename _CharT, typename _Traits, typename _Alloc> std::basic_string< _CharT, _Traits, _Alloc >::basic_string () [inline]`

Default constructor creates an empty string.

Definition at line 2957 of file `basic_string.h`.

Referenced by `std::basic_string< char >::basic_string()`, `std::basic_string< _CharT, _Traits, _Alloc >::basic_string()`, and `std::basic_string< char >::substr()`.

5.624.2.2 `template<typename _CharT, typename _Traits, typename _Alloc> std::basic_string< _CharT, _Traits, _Alloc >::basic_string (const _Alloc & __a) [explicit]`

Construct an empty string using allocator *a*.

Definition at line 618 of file `basic_string.tcc`.

References `std::basic_string< _CharT, _Traits, _Alloc >::basic_string()`.

5.624.2.3 `template<typename _CharT, typename _Traits, typename _Alloc> std::basic_string< _CharT, _Traits, _Alloc >::basic_string (const basic_string< _CharT, _Traits, _Alloc > & __str)`

Construct string with copy of value of *str*.

Parameters

<code>__str</code>	Source string.
--------------------	----------------

Definition at line 610 of file `basic_string.tcc`.

References `std::basic_string< _CharT, _Traits, _Alloc >::basic_string()`.

5.624.2.4 `template<typename _CharT, typename _Traits, typename _Alloc> std::basic_string< _CharT, _Traits, _Alloc >::basic_string (const basic_string< _CharT, _Traits, _Alloc > & __str, size_type __pos, size_type __n = npos)`

Construct string as copy of a substring.

Parameters

<code>__str</code>	Source string.
<code>__pos</code>	Index of first character to copy from.
<code>__n</code>	Number of characters to copy (default remainder).

Definition at line 624 of file `basic_string.tcc`.

References `std::basic_string< _CharT, _Traits, _Alloc >::basic_string()`.

5.624.2.5 `template<typename _CharT, typename _Traits, typename _Alloc> std::basic_string<_CharT, _Traits, _Alloc>::basic_string (const basic_string<_CharT, _Traits, _Alloc> & __str, size_type __pos, size_type __n, const _Alloc & __a)`

Construct string as copy of a substring.

Parameters

<code>__str</code>	Source string.
<code>__pos</code>	Index of first character to copy from.
<code>__n</code>	Number of characters to copy.
<code>__a</code>	Allocator to use.

Definition at line 634 of file `basic_string.tcc`.

References `std::basic_string<_CharT, _Traits, _Alloc>::basic_string()`.

5.624.2.6 `template<typename _CharT, typename _Traits, typename _Alloc> std::basic_string<_CharT, _Traits, _Alloc>::basic_string (const _CharT * __s, size_type __n, const _Alloc & __a = _Alloc())`

Construct string initialized by a character array.

Parameters

<code>__s</code>	Source character array.
<code>__n</code>	Number of characters to copy.
<code>__a</code>	Allocator to use (default is default allocator).

NB: `__s` must have at least `__n` characters, `'\0'` has no special meaning.

Definition at line 646 of file `basic_string.tcc`.

References `std::basic_string<_CharT, _Traits, _Alloc>::basic_string()`.

5.624.2.7 `template<typename _CharT, typename _Traits, typename _Alloc> std::basic_string<_CharT, _Traits, _Alloc>::basic_string (const _CharT * __s, const _Alloc & __a = _Alloc())`

Construct string as copy of a C string.

Parameters

<code>__s</code>	Source C string.
<code>__a</code>	Allocator to use (default is default allocator).

Definition at line 653 of file basic_string.tcc.

References `std::basic_string<_CharT, _Traits, _Alloc>::basic_string()`.

5.624.2.8 `template<typename _CharT, typename _Traits, typename _Alloc> std::basic_string<_CharT, _Traits, _Alloc>::basic_string (size_type __n, _CharT __c, const _Alloc & __a = _Alloc())`

Construct string as multiple characters.

Parameters

<code>↔ _n</code>	Number of characters.
<code>↔ _c</code>	Character to use.
<code>↔ _a</code>	Allocator to use (default is default allocator).

Definition at line 660 of file basic_string.tcc.

References `std::basic_string<_CharT, _Traits, _Alloc>::basic_string()`.

5.624.2.9 `template<typename _CharT, typename _Traits, typename _Alloc> std::basic_string<_CharT, _Traits, _Alloc>::basic_string (basic_string<_CharT, _Traits, _Alloc> && __str) [inline], [noexcept]`

Move construct string.

Parameters

<code>__str</code>	Source string.
--------------------	----------------

The newly-created string contains the exact contents of `__str`. `__str` is a valid, but unspecified string.

Definition at line 3027 of file basic_string.h.

5.624.2.10 `template<typename _CharT, typename _Traits, typename _Alloc> std::basic_string<_CharT, _Traits, _Alloc>::basic_string (initializer_list<_CharT> __l, const _Alloc & __a = _Alloc())`

Construct string from an initializer list.

Parameters

<code>↔ _l</code>	<code>std::initializer_list</code> of characters.
<code>↔ _a</code>	Allocator to use (default is default allocator).

Definition at line 675 of file basic_string.tcc.

References std::basic_string<_CharT, _Traits, _Alloc>::assign().

5.624.2.11 `template<typename _CharT, typename _Traits, typename _Alloc> template<typename _InputIterator >
std::basic_string<_CharT, _Traits, _Alloc>::basic_string (_InputIterator __beg, _InputIterator __end, const
_Alloc & __a = _Alloc())`

Construct string as copy of a range.

Parameters

<code>__beg</code>	Start of range.
<code>__end</code>	End of range.
<code>__a</code>	Allocator to use (default is default allocator).

Definition at line 668 of file basic_string.tcc.

References std::basic_string<_CharT, _Traits, _Alloc>::basic_string().

5.624.2.12 `template<typename _CharT, typename _Traits, typename _Alloc> std::basic_string<_CharT, _Traits, _Alloc
>::~basic_string() [inline], [noexcept]`

Destroy the string instance.

Definition at line 3061 of file basic_string.h.

5.624.3 Member Function Documentation

5.624.3.1 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string<_CharT, _Traits, _Alloc> &
std::basic_string<_CharT, _Traits, _Alloc>::append (const basic_string<_CharT, _Traits, _Alloc> & __str)`

Append a string to this string.

Parameters

<code>__str</code>	The string to append.
--------------------	-----------------------

Returns

Reference to this string.

Definition at line 765 of file basic_string.tcc.

References std::basic_string<_CharT, _Traits, _Alloc>::capacity(), std::basic_string<_CharT, _Traits, _Alloc>::reserve(), and std::basic_string<_CharT, _Traits, _Alloc>::size().

Referenced by `std::basic_string< char >::append()`, `std::basic_string< _CharT, _Traits, _Alloc >::append()`, `std::basic_string< _CharT, _Traits, _Alloc >::assign()`, `std::money_get< _CharT, _InIter >::do_get()`, `std::collate< _CharT >::do_transform()`, `std::basic_string< char >::operator+=()`, `std::operator>>()`, and `std::basic_string< _CharT, _Traits, _Alloc >::resize()`.

5.624.3.2 `template<typename _CharT, typename _Traits, typename _Alloc > basic_string< _CharT, _Traits, _Alloc > & std::basic_string< _CharT, _Traits, _Alloc >::append (const basic_string< _CharT, _Traits, _Alloc > & __str, size_type __pos, size_type __n)`

Append a substring.

Parameters

<code>__str</code>	The string to append.
<code>__pos</code>	Index of the first character of <code>str</code> to append.
<code>__n</code>	The number of characters to append.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	if <code>__pos</code> is not a valid index.
--------------------------------	---

This function appends `__n` characters from `__str` starting at `__pos` to this string. If `__n` is larger than the number of available characters in `__str`, the remainder of `__str` is appended.

Definition at line 782 of file `basic_string.tcc`.

References `std::basic_string< _CharT, _Traits, _Alloc >::capacity()`, `std::basic_string< _CharT, _Traits, _Alloc >::insert()`, `std::basic_string< _CharT, _Traits, _Alloc >::reserve()`, and `std::basic_string< _CharT, _Traits, _Alloc >::size()`.

5.624.3.3 `template<typename _CharT, typename _Traits, typename _Alloc > basic_string< _CharT, _Traits, _Alloc > & std::basic_string< _CharT, _Traits, _Alloc >::append (const _CharT * __s, size_type __n)`

Append a C substring.

Parameters

<code>__s</code>	The C string to append.
<code>__n</code>	The number of characters to append.

Returns

Reference to this string.

Definition at line 738 of file basic_string.tcc.

References std::basic_string< _CharT, _Traits, _Alloc >::append(), std::basic_string< _CharT, _Traits, _Alloc >::capacity(), std::basic_string< _CharT, _Traits, _Alloc >::reserve(), and std::basic_string< _CharT, _Traits, _Alloc >::size().

5.624.3.4 template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string< _CharT, _Traits, _Alloc >::append (const _CharT * __s) [inline]

Append a C string.

Parameters

<code>__s</code>	The C string to append.
------------------	-------------------------

Returns

Reference to this string.

Definition at line 3552 of file basic_string.h.

5.624.3.5 template<typename _CharT, typename _Traits, typename _Alloc > basic_string< _CharT, _Traits, _Alloc > & std::basic_string< _CharT, _Traits, _Alloc >::append (size_type __n, _CharT __c)

Append multiple characters.

Parameters

<code>__n</code>	The number of characters to append.
<code>__c</code>	The character to use.

Returns

Reference to this string.

Appends __n copies of __c to this string.

Definition at line 721 of file basic_string.tcc.

References std::basic_string< _CharT, _Traits, _Alloc >::append(), std::basic_string< _CharT, _Traits, _Alloc >::capacity(), std::basic_string< _CharT, _Traits, _Alloc >::reserve(), and std::basic_string< _CharT, _Traits, _Alloc >::size().

5.624.3.6 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string< _CharT, _Traits, _Alloc >::append (initializer_list< _CharT > __l) [inline]`

Append an initializer_list of characters.

Parameters

↩	The initializer_list of characters to append.
__↩	
↩	
__↩	
<i>l</i>	

Returns

Reference to this string.

Definition at line 3576 of file basic_string.h.

5.624.3.7 `template<typename _CharT, typename _Traits, typename _Alloc> template<class _InputIterator > basic_string& std::basic_string< _CharT, _Traits, _Alloc >::append (_InputIterator __first, _InputIterator __last) [inline]`

Append a range of characters.

Parameters

__first	Iterator referencing the first character to append.
__last	Iterator marking the end of the range.

Returns

Reference to this string.

Appends characters in the range [__first,__last) to this string.

Definition at line 3590 of file basic_string.h.

5.624.3.8 `template<typename _CharT , typename _Traits , typename _Alloc > basic_string< _CharT, _Traits, _Alloc > & std::basic_string< _CharT, _Traits, _Alloc >::assign (const basic_string< _CharT, _Traits, _Alloc > & __str)`

Set value to contents of another string.

Parameters

__str	Source string to use.
-------	-----------------------

Returns

Reference to this string.

Definition at line 683 of file basic_string.tcc.

References std::basic_string< _CharT, _Traits, _Alloc >::get_allocator().

Referenced by std::basic_string< char >::assign(), std::basic_string< _CharT, _Traits, _Alloc >::basic_string(), std::basic_string< _CharT, _Traits, _Alloc >::do_get(), std::basic_string< char >::operator=(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow(), and std::basic_string< char >::push_back().

5.624.3.9 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string< _CharT, _Traits, _Alloc >::assign (basic_string< _CharT, _Traits, _Alloc > && __str) [inline]`

Set value to contents of another string.

Parameters

<code>__str</code>	Source string to use.
--------------------	-----------------------

Returns

Reference to this string.

This function sets this string to the exact contents of `__str`. `__str` is a valid, but unspecified string.

Definition at line 3626 of file basic_string.h.

5.624.3.10 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string< _CharT, _Traits, _Alloc >::assign (const basic_string< _CharT, _Traits, _Alloc > & __str, size_type __pos, size_type __n) [inline]`

Set value to a substring of a string.

Parameters

<code>__str</code>	The string to use.
<code>__pos</code>	Index of the first character of str.
<code>__n</code>	Number of characters to use.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	if <code>pos</code> is not a valid index.
--------------------------------	---

This function sets this string to the substring of `__str` consisting of `__n` characters at `__pos`. If `__n` is larger than the number of available characters in `__str`, the remainder of `__str` is used.

Definition at line 3647 of file `basic_string.h`.

5.624.3.11 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string<_CharT, _Traits, _Alloc> &std::basic_string<_CharT, _Traits, _Alloc>::assign (const _CharT * __s, size_type __n)`

Set value to a C substring.

Parameters

<code>__s</code>	The C string to use.
<code>__n</code>	Number of characters to use.

Returns

Reference to this string.

This function sets the value of this string to the first `__n` characters of `__s`. If `__n` is larger than the number of available characters in `__s`, the remainder of `__s` is used.

Definition at line 699 of file `basic_string.tcc`.

References `std::basic_string<_CharT, _Traits, _Alloc>::append()`, and `std::basic_string<_CharT, _Traits, _Alloc>::size()`.

5.624.3.12 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string<_CharT, _Traits, _Alloc>::assign (const _CharT * __s) [inline]`

Set value to contents of a C string.

Parameters

<code>__s</code>	The C string to use.
------------------	----------------------

Returns

Reference to this string.

This function sets the value of this string to the value of `__s`. The data is copied, so there is no dependence on `__s` once the function returns.

Definition at line 3675 of file `basic_string.h`.

5.624.3.13 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string<_CharT, _Traits, _Alloc>::assign (size_type __n, _CharT __c) [inline]`

Set value to multiple characters.

Parameters

<code>__n</code>	Length of the resulting string.
<code>__c</code>	The character to use.

Returns

Reference to this string.

This function sets the value of this string to `__n` copies of character `__c`.

Definition at line 3691 of file `basic_string.h`.

5.624.3.14 `template<typename _CharT, typename _Traits, typename _Alloc> template<class _InputIterator> basic_string& std::basic_string<_CharT, _Traits, _Alloc>::assign (_InputIterator __first, _InputIterator __last) [inline]`

Set value to a range of characters.

Parameters

<code>__first</code>	Iterator referencing the first character to append.
<code>__last</code>	Iterator marking the end of the range.

Returns

Reference to this string.

Sets value of string to characters in the range `[__first, __last)`.

Definition at line 3704 of file `basic_string.h`.

5.624.3.15 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string<_CharT, _Traits, _Alloc>::assign (initializer_list<_CharT> __l) [inline]`

Set value to an `initializer_list` of characters.

Parameters

\leftrightarrow	The initializer_list of characters to assign.
$_ \leftrightarrow$	
\leftarrow	
$_ \leftrightarrow$	
$/$	

Returns

Reference to this string.

Definition at line 3714 of file basic_string.h.

5.624.3.16 `template<typename _CharT, typename _Traits, typename _Alloc> const_reference std::basic_string< _CharT, _Traits, _Alloc >::at (size_type __n) const [inline]`

Provides access to the data contained in the string.

Parameters

$_ \leftrightarrow$	The index of the character to access.
$_n$	

Returns

Read-only (const) reference to the character.

Exceptions

<code>std::out_of_range</code>	If n is an invalid index.
--------------------------------	-----------------------------

This function provides for safer data access. The parameter is first checked that it is in the range of the string. The function throws `out_of_range` if the check fails.

Definition at line 3392 of file basic_string.h.

5.624.3.17 `template<typename _CharT, typename _Traits, typename _Alloc> reference std::basic_string< _CharT, _Traits, _Alloc >::at (size_type __n) [inline]`

Provides access to the data contained in the string.

Parameters

$_ \leftrightarrow$	The index of the character to access.
$_n$	

Returns

Read/write reference to the character.

Exceptions

<code>std::out_of_range</code>	If <i>n</i> is an invalid index.
--------------------------------	----------------------------------

This function provides for safer data access. The parameter is first checked that it is in the range of the string. The function throws `out_of_range` if the check fails. Success results in unsharing the string.

Definition at line 3414 of file `basic_string.h`.

5.624.3.18 `template<typename _CharT, typename _Traits, typename _Alloc> reference std::basic_string< _CharT, _Traits, _Alloc >::back () [inline]`

Returns a read/write reference to the data at the last element of the string.

Definition at line 3453 of file `basic_string.h`.

5.624.3.19 `template<typename _CharT, typename _Traits, typename _Alloc> const_reference std::basic_string< _CharT, _Traits, _Alloc >::back () const [inline], [noexcept]`

Returns a read-only (constant) reference to the data at the last element of the string.

Definition at line 3464 of file `basic_string.h`.

5.624.3.20 `template<typename _CharT, typename _Traits, typename _Alloc> iterator std::basic_string< _CharT, _Traits, _Alloc >::begin () [inline]`

Returns a read/write iterator that points to the first character in the string. Unshares the string.

Definition at line 3129 of file `basic_string.h`.

Referenced by `__gnu_profile::__report()`, `std::basic_string< char >::crend()`, `std::regex_match()`, `std::regex_replace()`, `std::regex_search()`, and `std::basic_string< char >::rend()`.

5.624.3.21 `template<typename _CharT, typename _Traits, typename _Alloc> const_iterator std::basic_string< _CharT, _Traits, _Alloc >::begin () const [inline], [noexcept]`

Returns a read-only (constant) iterator that points to the first character in the string.

Definition at line 3140 of file `basic_string.h`.

5.624.3.22 `template<typename _CharT, typename _Traits, typename _Alloc> const _CharT* std::basic_string< _CharT, _Traits, _Alloc >::c_str () const [inline], [noexcept]`

Return const pointer to null-terminated contents.

This is a handle to internal data. Do not modify or dire things may happen.

Definition at line 4351 of file basic_string.h.

Referenced by `__gnu_profile::__report()`, `std::collate< _CharT >::do_compare()`, `std::money_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::collate< _CharT >::do_transform()`, `std::getline()`, `std::messages< _CharT >::messages()`, `std::regex_replace()`, and `std::messages< _CharT >::~~messages()`.

5.624.3.23 `template<typename _CharT, typename _Traits, typename _Alloc> size_type std::basic_string< _CharT, _Traits, _Alloc >::capacity () const [inline], [noexcept]`

Returns the total number of characters that the string can hold before needing to allocate more memory.

Definition at line 3302 of file basic_string.h.

Referenced by `std::basic_string< _CharT, _Traits, _Alloc >::append()`, `std::basic_string< char >::push_back()`, `std::basic_string< _CharT, _Traits, _Alloc >::replace()`, `std::basic_string< _CharT, _Traits, _Alloc >::reserve()`, and `std::basic_string< char >::shrink_to_fit()`.

5.624.3.24 `template<typename _CharT, typename _Traits, typename _Alloc> const_iterator std::basic_string< _CharT, _Traits, _Alloc >::cbegin () const [inline], [noexcept]`

Returns a read-only (constant) iterator that points to the first character in the string.

Definition at line 3204 of file basic_string.h.

5.624.3.25 `template<typename _CharT, typename _Traits, typename _Alloc> const_iterator std::basic_string< _CharT, _Traits, _Alloc >::cend () const [inline], [noexcept]`

Returns a read-only (constant) iterator that points one past the last character in the string.

Definition at line 3212 of file basic_string.h.

5.624.3.26 `template<typename _CharT, typename _Traits, typename _Alloc> void std::basic_string< _CharT, _Traits, _Alloc >::clear () [inline]`

Erases the string, making it empty.

Definition at line 3330 of file basic_string.h.

5.624.3.27 `template<typename _CharT, typename _Traits, typename _Alloc> int std::basic_string< _CharT, _Traits, _Alloc >::compare (const basic_string< _CharT, _Traits, _Alloc > & __str) const [inline]`

Compare to a string.

Parameters

<code>__str</code>	String to compare against.
--------------------	----------------------------

Returns

Integer < 0, 0, or > 0.

Returns an integer < 0 if this string is ordered before `__str`, 0 if their values are equivalent, or > 0 if this string is ordered after `__str`. Determines the effective length `rlen` of the strings to compare as the smallest of `size()` and `str.size()`. The function then compares the two strings by calling `traits::compare(data(), str.data(), rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

Definition at line 4776 of file `basic_string.h`.

Referenced by `std::basic_string< char >::compare()`, `std::basic_string< _CharT, _Traits, _Alloc >::compare()`, `std::basic_string< _CharT, _Traits, _Alloc >::find_last_not_of()`, `std::operator<()`, and `std::operator<=()`.

5.624.3.28 `template<typename _CharT, typename _Traits, typename _Alloc> int std::basic_string< _CharT, _Traits, _Alloc >::compare (size_type __pos, size_type __n, const basic_string< _CharT, _Traits, _Alloc > & __str) const`

Compare substring to a string.

Parameters

<code>__pos</code>	Index of first character of substring.
<code>__n</code>	Number of characters in substring.
<code>__str</code>	String to compare against.

Returns

Integer < 0, 0, or > 0.

Form the substring of this string from the `__n` characters starting at `__pos`. Returns an integer < 0 if the substring is ordered before `__str`, 0 if their values are equivalent, or > 0 if the substring is ordered after `__str`. Determines the effective length `rlen` of the strings to compare as the smallest of the length of the substring and `__str.size()`. The function then compares the two strings by calling `traits::compare(substring.data(), str.data(), rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

Definition at line 1357 of file `basic_string.tcc`.

References `std::basic_string< _CharT, _Traits, _Alloc >::compare()`, `std::basic_string< _CharT, _Traits, _Alloc >::data()`, `std::min()`, and `std::basic_string< _CharT, _Traits, _Alloc >::size()`.

5.624.3.29 `template<typename _CharT, typename _Traits, typename _Alloc> int std::basic_string< _CharT, _Traits, _Alloc >::compare (size_type __pos1, size_type __n1, const basic_string< _CharT, _Traits, _Alloc > & __str, size_type __pos2, size_type __n2) const`

Compare substring to a substring.

Parameters

<code>__pos1</code>	Index of first character of substring.
<code>__n1</code>	Number of characters in substring.
<code>__str</code>	String to compare against.
<code>__pos2</code>	Index of first character of substring of <code>str</code> .
<code>__n2</code>	Number of characters in substring of <code>str</code> .

Returns

Integer < 0 , 0 , or > 0 .

Form the substring of this string from the `__n1` characters starting at `__pos1`. Form the substring of `__str` from the `__n2` characters starting at `__pos2`. Returns an integer < 0 if this substring is ordered before the substring of `__str`, 0 if their values are equivalent, or > 0 if this substring is ordered after the substring of `__str`. Determines the effective length `rlen` of the strings to compare as the smallest of the lengths of the substrings. The function then compares the two strings by calling `traits::compare(substring.data(),str.substr(pos2,n2).data(),rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

Definition at line 1372 of file `basic_string.tcc`.

References `std::basic_string<_CharT, _Traits, _Alloc>::compare()`, `std::basic_string<_CharT, _Traits, _Alloc>::data()`, and `std::min()`.

5.624.3.30 `template<typename _CharT, typename _Traits, typename _Alloc> int std::basic_string<_CharT, _Traits, _Alloc>::compare (const _CharT * __s) const`

Compare to a C string.

Parameters

<code>__s</code>	C string to compare against.
------------------	------------------------------

Returns

Integer < 0 , 0 , or > 0 .

Returns an integer < 0 if this string is ordered before `__s`, 0 if their values are equivalent, or > 0 if this string is ordered after `__s`. Determines the effective length `rlen` of the strings to compare as the smallest of `size()` and the length of a string constructed from `__s`. The function then compares the two strings by calling `traits::compare(data(),s,rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

Definition at line 1390 of file `basic_string.tcc`.

References `std::min()`, and `std::basic_string<_CharT, _Traits, _Alloc>::size()`.

5.624.3.31 `template<typename _CharT, typename _Traits, typename _Alloc> int std::basic_string<_CharT, _Traits, _Alloc>::compare (size_type __pos, size_type __n1, const _CharT * __s) const`

Compare substring to a C string.

Parameters

<code>__pos</code>	Index of first character of substring.
<code>__n1</code>	Number of characters in substring.
<code>__s</code>	C string to compare against.

Returns

Integer < 0, 0, or > 0.

Form the substring of this string from the `__n1` characters starting at `pos`. Returns an integer < 0 if the substring is ordered before `__s`, 0 if their values are equivalent, or > 0 if the substring is ordered after `__s`. Determines the effective length `rlen` of the strings to compare as the smallest of the length of the substring and the length of a string constructed from `__s`. The function then compares the two string by calling `traits::compare(substring.data(), __s, rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

Definition at line 1405 of file `basic_string.tcc`.

References `std::min()`.

5.624.3.32 `template<typename _CharT, typename _Traits, typename _Alloc> int std::basic_string<_CharT, _Traits, _Alloc>::compare(size_type __pos, size_type __n1, const _CharT * __s, size_type __n2) const`

Compare substring against a character array.

Parameters

<code>__pos</code>	Index of first character of substring.
<code>__n1</code>	Number of characters in substring.
<code>__s</code>	character array to compare against.
<code>__n2</code>	Number of characters of <code>s</code> .

Returns

Integer < 0, 0, or > 0.

Form the substring of this string from the `__n1` characters starting at `__pos`. Form a string from the first `__n2` characters of `__s`. Returns an integer < 0 if this substring is ordered before the string from `__s`, 0 if their values are equivalent, or > 0 if this substring is ordered after the string from `__s`. Determines the effective length `rlen` of the strings to compare as the smallest of the length of the substring and `__n2`. The function then compares the two strings by calling `traits::compare(substring.data(), s, rlen)`. If the result of the comparison is nonzero returns it, otherwise the shorter one is ordered first.

NB: `s` must have at least `n2` characters, `'\0'` has no special meaning.

Definition at line 1421 of file `basic_string.tcc`.

References `std::min()`.

5.624.3.33 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string<_CharT, _Traits, _Alloc>::size_type
std::basic_string<_CharT, _Traits, _Alloc>::copy (_CharT * __s, size_type __n, size_type __pos = 0) const`

Copy substring into C string.

Parameters

<code>__s</code>	C string to copy value into.
<code>__n</code>	Number of characters to copy.
<code>__pos</code>	Index of first character to copy.

Returns

Number of characters actually copied

Exceptions

<code>std::out_of_range</code>	If <code>__pos > size()</code> .
--------------------------------	-------------------------------------

Copies up to `__n` characters starting at `__pos` into the C string `__s`. If `__pos` is greater than `size()`, `out_of_range` is thrown.

Definition at line 1133 of file `basic_string.tcc`.

Referenced by `std::basic_string<char>::replace()`, and `std::basic_string<_CharT, _Traits, _Alloc>::resize()`.

5.624.3.34 `template<typename _CharT, typename _Traits, typename _Alloc> const_reverse_iterator std::basic_string<
_CharT, _Traits, _Alloc>::crbegin () const [inline], [noexcept]`

Returns a read-only (constant) reverse iterator that points to the last character in the string. Iteration is done in reverse element order.

Definition at line 3221 of file `basic_string.h`.

5.624.3.35 `template<typename _CharT, typename _Traits, typename _Alloc> const_reverse_iterator std::basic_string<
_CharT, _Traits, _Alloc>::crend () const [inline], [noexcept]`

Returns a read-only (constant) reverse iterator that points to one before the first character in the string. Iteration is done in reverse element order.

Definition at line 3230 of file `basic_string.h`.

5.624.3.36 `template<typename _CharT, typename _Traits, typename _Alloc> const _CharT* std::basic_string< _CharT, _Traits, _Alloc >::data () const [inline], [noexcept]`

Return const pointer to contents.

This is a handle to internal data. Do not modify or dire things may happen.

Definition at line 4361 of file basic_string.h.

Referenced by std::basic_regex< _Ch_type, _Rx_traits >::assign(), std::locale::combine(), std::basic_string< _CharT, _Traits, _Alloc >::compare(), std::collate< _CharT >::do_compare(), std::money_get< _CharT, _InIter >::do_get(), std::collate< _CharT >::do_transform(), std::match_results< _Bi_iter >::format(), std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::from_bytes(), std::operator<<(), std::operator==(), std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::to_bytes(), std::regex_traits< _Ch_type >::transform(), and std::experimental::filesystem::v1::u8path().

5.624.3.37 `template<typename _CharT, typename _Traits, typename _Alloc> bool std::basic_string< _CharT, _Traits, _Alloc >::empty () const [inline], [noexcept]`

Returns true if the string is empty. Equivalent to `*this == ""`.

Definition at line 3338 of file basic_string.h.

Referenced by std::basic_string< char >::back(), std::basic_string< char >::front(), std::tr2::operator>>(), and std::basic_string< char >::pop_back().

5.624.3.38 `template<typename _CharT, typename _Traits, typename _Alloc> iterator std::basic_string< _CharT, _Traits, _Alloc >::end () [inline]`

Returns a read/write iterator that points one past the last character in the string. Unshares the string.

Definition at line 3148 of file basic_string.h.

Referenced by __gnu_profile::__report(), std::basic_string< char >::crbegin(), std::basic_string< char >::rbegin(), std::regex_match(), std::regex_replace(), and std::regex_search().

5.624.3.39 `template<typename _CharT, typename _Traits, typename _Alloc> const_iterator std::basic_string< _CharT, _Traits, _Alloc >::end () const [inline], [noexcept]`

Returns a read-only (constant) iterator that points one past the last character in the string.

Definition at line 3159 of file basic_string.h.

5.624.3.40 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string< _CharT, _Traits, _Alloc >::erase (size_type __pos = 0, size_type __n = npos) [inline]`

Remove characters.

Parameters

<code>__pos</code>	Index of first character to remove (default 0).
<code>__n</code>	Number of characters to remove (default remainder).

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <i>pos</i> is beyond the end of this string.
--------------------------------	---

Removes `__n` characters from this string starting at `__pos`. The length of the string is reduced by `__n`. If there are `< __n` characters to remove, the remainder of the string is truncated. If `__p` is beyond end of string, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 3909 of file `basic_string.h`.

Referenced by `__gnu_profile::__report()`, `std::money_get<_CharT, _InIter >::do_get()`, `std::basic_string<char>::erase()`, `std::getline()`, `std::basic_string<_CharT, _Traits, _Alloc>::insert()`, `std::operator>>()`, `std::basic_string<char>::pop_back()`, and `std::basic_string<_CharT, _Traits, _Alloc>::resize()`.

5.624.3.41 `template<typename _CharT, typename _Traits, typename _Alloc> iterator std::basic_string<_CharT, _Traits, _Alloc>::erase (iterator __position) [inline]`

Remove one character.

Parameters

<code>__position</code>	Iterator referencing the character to remove.
-------------------------	---

Returns

iterator referencing same location after removal.

Removes the character at `__position` from this string. The value of the string doesn't change if an error is thrown.

Definition at line 3925 of file `basic_string.h`.

5.624.3.42 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string<_CharT, _Traits, _Alloc>::iterator std::basic_string<_CharT, _Traits, _Alloc>::erase (iterator __first, iterator __last)`

Remove a range of characters.

Parameters

<code>__first</code>	Iterator referencing the first character to remove.
<code>__last</code>	Iterator referencing the end of the range.

Returns

Iterator referencing location of first after removal.

Removes the characters in the range [first,last) from this string. The value of the string doesn't change if an error is thrown.

Definition at line 831 of file basic_string.tcc.

References std::basic_string< _CharT, _Traits, _Alloc >::replace().

5.624.3.43 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string< _CharT, _Traits, _Alloc >::size_type
std::basic_string< _CharT, _Traits, _Alloc >::find (const _CharT * __s, size_type __pos, size_type __n) const`

Find position of a C substring.

Parameters

<code>__s</code>	C string to locate.
<code>__pos</code>	Index of character to search from.
<code>__n</code>	Number of characters from s to search for.

Returns

Index of start of first occurrence.

Starting from `__pos`, searches forward for the first `__n` characters in `__s` within this string. If found, returns the index where it begins. If not found, returns npos.

Definition at line 1178 of file basic_string.tcc.

References std::basic_string< _CharT, _Traits, _Alloc >::npos, and std::basic_string< _CharT, _Traits, _Alloc >::size().

Referenced by __gnu_profile::__report(), std::basic_string< char >::find(), std::basic_string< char >::find_first_of(), std::basic_string< char >::get_allocator(), and std::operator+().

5.624.3.44 `template<typename _CharT, typename _Traits, typename _Alloc> size_type std::basic_string< _CharT, _Traits,
_Alloc >::find (const basic_string< _CharT, _Traits, _Alloc > & __str, size_type __pos = 0) const [inline],
[noexcept]`

Find position of a string.

Parameters

<code>__str</code>	String to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of start of first occurrence.

Starting from `__pos`, searches forward for value of `__str` within this string. If found, returns the index where it begins. If not found, returns `npos`.

Definition at line 4397 of file `basic_string.h`.

```
5.624.3.45  template<typename _CharT, typename _Traits, typename _Alloc> size_type std::basic_string< _CharT, _Traits,
            _Alloc >::find ( const _CharT * __s, size_type __pos = 0 ) const    [inline]
```

Find position of a C string.

Parameters

<code>__s</code>	C string to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of start of first occurrence.

Starting from `__pos`, searches forward for the value of `__s` within this string. If found, returns the index where it begins. If not found, returns `npos`.

Definition at line 4412 of file `basic_string.h`.

```
5.624.3.46  template<typename _CharT, typename _Traits, typename _Alloc > basic_string< _CharT, _Traits, _Alloc >::size_type
            std::basic_string< _CharT, _Traits, _Alloc >::find ( _CharT __c, size_type __pos = 0 ) const    [noexcept]
```

Find position of a character.

Parameters

<code>__c</code>	Character to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 1201 of file `basic_string.tcc`.

References `std::basic_string< _CharT, _Traits, _Alloc >::npos`, `std::basic_string< _CharT, _Traits, _Alloc >::rfind()`, and `std::basic_string< _CharT, _Traits, _Alloc >::size()`.

5.624.3.47 `template<typename _CharT, typename _Traits, typename _Alloc> size_type std::basic_string< _CharT, _Traits, _Alloc >::find_first_not_of (const basic_string< _CharT, _Traits, _Alloc > & __str, size_type __pos = 0) const [inline], [noexcept]`

Find position of a character not in string.

Parameters

<code>__str</code>	String containing characters to avoid.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for a character not contained in `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 4630 of file `basic_string.h`.

Referenced by `__gnu_profile::__report()`, `std::basic_string< char >::find_first_not_of()`, `std::basic_string< _CharT, _Traits, _Alloc >::find_first_not_of()`, and `std::basic_string< _CharT, _Traits, _Alloc >::find_last_of()`.

5.624.3.48 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string< _CharT, _Traits, _Alloc >::size_type std::basic_string< _CharT, _Traits, _Alloc >::find_first_not_of (const _CharT * __s, size_type __pos, size_type __n) const`

Find position of a character not in C substring.

Parameters

<code>__s</code>	C string containing characters to avoid.
<code>__pos</code>	Index of character to search from.
<code>__n</code>	Number of characters from <code>__s</code> to consider.

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for a character not contained in the first `__n` characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 1293 of file basic_string.tcc.

References `std::basic_string<_CharT, _Traits, _Alloc>::find_first_not_of()`, `std::basic_string<_CharT, _Traits, _Alloc>::npos`, and `std::basic_string<_CharT, _Traits, _Alloc>::size()`.

5.624.3.49 `template<typename _CharT, typename _Traits, typename _Alloc> size_type std::basic_string<_CharT, _Traits, _Alloc>::find_first_not_of (const _CharT* __s, size_type __pos = 0) const [inline]`

Find position of a character not in C string.

Parameters

<code>__s</code>	C string containing characters to avoid.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for a character not contained in `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 4661 of file basic_string.h.

5.624.3.50 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string<_CharT, _Traits, _Alloc>::size_type std::basic_string<_CharT, _Traits, _Alloc>::find_first_not_of (_CharT __c, size_type __pos = 0) const [noexcept]`

Find position of a different character.

Parameters

<code>__c</code>	Character to avoid.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for a character other than `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 1305 of file basic_string.tcc.

References `std::basic_string<_CharT, _Traits, _Alloc>::find_last_not_of()`, `std::basic_string<_CharT, _Traits, _Alloc>::npos`, and `std::basic_string<_CharT, _Traits, _Alloc>::size()`.

5.624.3.51 `template<typename _CharT, typename _Traits, typename _Alloc> size_type std::basic_string< _CharT, _Traits, _Alloc >::find_first_of(const basic_string< _CharT, _Traits, _Alloc > & __str, size_type __pos = 0) const [inline], [noexcept]`

Find position of a character of string.

Parameters

<code>__str</code>	String containing characters to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for one of the characters of `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 4503 of file `basic_string.h`.

Referenced by `__gnu_profile::__report()`, `std::basic_string< char >::find_first_of()`, and `std::basic_string< _CharT, _Traits, _Alloc >::rfind()`.

5.624.3.52 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string< _CharT, _Traits, _Alloc >::size_type std::basic_string< _CharT, _Traits, _Alloc >::find_first_of(const _CharT * __s, size_type __pos, size_type __n) const`

Find position of a character of C substring.

Parameters

<code>__s</code>	String containing characters to locate.
<code>__pos</code>	Index of character to search from.
<code>__n</code>	Number of characters from <code>s</code> to search for.

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for one of the first `__n` characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 1257 of file `basic_string.tcc`.

References `std::basic_string< _CharT, _Traits, _Alloc >::find_last_of()`, `std::basic_string< _CharT, _Traits, _Alloc >::npos`, and `std::basic_string< _CharT, _Traits, _Alloc >::size()`.

5.624.3.53 `template<typename _CharT, typename _Traits, typename _Alloc> size_type std::basic_string< _CharT, _Traits, _Alloc >::find_first_of(const _CharT * __s, size_type __pos = 0) const [inline]`

Find position of a character of C string.

Parameters

<code>__s</code>	String containing characters to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for one of the characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 4533 of file `basic_string.h`.

5.624.3.54 `template<typename _CharT, typename _Traits, typename _Alloc> size_type std::basic_string< _CharT, _Traits, _Alloc >::find_first_of(_CharT __c, size_type __pos = 0) const` `[inline]`, `[noexcept]`

Find position of a character.

Parameters

<code>__c</code>	Character to locate.
<code>__pos</code>	Index of character to search from (default 0).

Returns

Index of first occurrence.

Starting from `__pos`, searches forward for the character `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Note: equivalent to `find(__c, __pos)`.

Definition at line 4552 of file `basic_string.h`.

5.624.3.55 `template<typename _CharT, typename _Traits, typename _Alloc> size_type std::basic_string< _CharT, _Traits, _Alloc >::find_last_not_of(const basic_string< _CharT, _Traits, _Alloc > & __str, size_type __pos = npos) const` `[inline]`, `[noexcept]`

Find last position of a character not in string.

Parameters

<code>__str</code>	String containing characters to avoid.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for a character not contained in `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 4693 of file `basic_string.h`.

Referenced by `std::basic_string< _CharT, _Traits, _Alloc >::find_first_not_of()`, `std::basic_string< char >::find_last_not_of()`, and `std::basic_string< _CharT, _Traits, _Alloc >::find_last_not_of()`.

5.624.3.56 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string< _CharT, _Traits, _Alloc >::size_type std::basic_string< _CharT, _Traits, _Alloc >::find_last_not_of (const _CharT* __s, size_type __pos, size_type __n) const`

Find last position of a character not in C substring.

Parameters

<code>__s</code>	C string containing characters to avoid.
<code>__pos</code>	Index of character to search back from.
<code>__n</code>	Number of characters from <code>s</code> to consider.

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for a character not contained in the first `__n` characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 1316 of file `basic_string.tcc`.

References `std::basic_string< _CharT, _Traits, _Alloc >::find_last_not_of()`, `std::basic_string< _CharT, _Traits, _Alloc >::npos`, and `std::basic_string< _CharT, _Traits, _Alloc >::size()`.

5.624.3.57 `template<typename _CharT, typename _Traits, typename _Alloc> size_type std::basic_string< _CharT, _Traits, _Alloc >::find_last_not_of (const _CharT* __s, size_type __pos = npos) const [inline]`

Find last position of a character not in C string.

Parameters

<code>__s</code>	C string containing characters to avoid.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for a character not contained in `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 4724 of file `basic_string.h`.

```
5.624.3.58  template<typename _CharT, typename _Traits, typename _Alloc > basic_string< _CharT, _Traits, _Alloc
>::size_type std::basic_string< _CharT, _Traits, _Alloc >::find_last_not_of ( _CharT __c, size_type __pos = npos )
const      [noexcept]
```

Find last position of a different character.

Parameters

<code>__c</code>	Character to avoid.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for a character other than `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 1337 of file `basic_string.tcc`.

References `std::basic_string< _CharT, _Traits, _Alloc >::compare()`, `std::basic_string< _CharT, _Traits, _Alloc >::npos`, and `std::basic_string< _CharT, _Traits, _Alloc >::size()`.

```
5.624.3.59  template<typename _CharT, typename _Traits, typename _Alloc> size_type std::basic_string< _CharT, _Traits,
_Alloc >::find_last_of ( const basic_string< _CharT, _Traits, _Alloc > & __str, size_type __pos = npos ) const
[inline], [noexcept]
```

Find last position of a character of string.

Parameters

<code>__str</code>	String containing characters to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for one of the characters of `__str` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 4567 of file basic_string.h.

Referenced by std::basic_string< _CharT, _Traits, _Alloc >::find_first_of(), and std::basic_string< char >::find_last_of().

5.624.3.60 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string< _CharT, _Traits, _Alloc>::size_type std::basic_string< _CharT, _Traits, _Alloc>::find_last_of(const _CharT * __s, size_type __pos, size_type __n) const`

Find last position of a character of C substring.

Parameters

<code>__s</code>	C string containing characters to locate.
<code>__pos</code>	Index of character to search back from.
<code>__n</code>	Number of characters from s to search for.

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for one of the first `__n` characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 1272 of file basic_string.tcc.

References std::basic_string< _CharT, _Traits, _Alloc >::find_first_not_of(), std::basic_string< _CharT, _Traits, _Alloc >::npos, and std::basic_string< _CharT, _Traits, _Alloc >::size().

5.624.3.61 `template<typename _CharT, typename _Traits, typename _Alloc> size_type std::basic_string< _CharT, _Traits, _Alloc>::find_last_of(const _CharT * __s, size_type __pos = npos) const [inline]`

Find last position of a character of C string.

Parameters

<code>__s</code>	C string containing characters to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for one of the characters of `__s` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 4597 of file basic_string.h.

5.624.3.62 `template<typename _CharT, typename _Traits, typename _Alloc> size_type std::basic_string< _CharT, _Traits, _Alloc >::find_last_of (_CharT __c, size_type __pos = npos) const` `[inline], [noexcept]`

Find last position of a character.

Parameters

<code>__c</code>	Character to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Note: equivalent to `rfind(__c, __pos)`.

Definition at line 4616 of file `basic_string.h`.

5.624.3.63 `template<typename _CharT, typename _Traits, typename _Alloc> reference std::basic_string< _CharT, _Traits, _Alloc >::front ()` `[inline]`

Returns a read/write reference to the data at the first element of the string.

Definition at line 3431 of file `basic_string.h`.

5.624.3.64 `template<typename _CharT, typename _Traits, typename _Alloc> const_reference std::basic_string< _CharT, _Traits, _Alloc >::front () const` `[inline], [noexcept]`

Returns a read-only (constant) reference to the data at the first element of the string.

Definition at line 3442 of file `basic_string.h`.

5.624.3.65 `template<typename _CharT, typename _Traits, typename _Alloc> allocator_type std::basic_string< _CharT, _Traits, _Alloc >::get_allocator () const` `[inline], [noexcept]`

Return copy of allocator used to construct this string.

Definition at line 4368 of file `basic_string.h`.

Referenced by `std::basic_string< _CharT, _Traits, _Alloc >::assign()`, `std::basic_string< char >::basic_string()`, `std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::from_bytes()`, `std::basic_string< _CharT, _Traits, _Alloc >::replace()`, `std::basic_string< _CharT, _Traits, _Alloc >::reserve()`, `std::basic_string< _CharT, _Traits, _Alloc >::swap()`, `std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::to_bytes()`, and `std::basic_string< char >::~~basic_string()`.

5.624.3.66 `template<typename _CharT, typename _Traits, typename _Alloc> void std::basic_string< _CharT, _Traits, _Alloc >::insert (iterator __p, size_type __n, _CharT __c)` `[inline]`

Insert multiple characters.

Parameters

<code>__p</code>	Iterator referencing location in string to insert at.
<code>__n</code>	Number of characters to insert
<code>__c</code>	The character to insert.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts `__n` copies of character `__c` starting at the position referenced by iterator `__p`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 3732 of file `basic_string.h`.

Referenced by `std::basic_string< _CharT, _Traits, _Alloc >::append()`, `std::money_get< _CharT, _InIter >::do_get()`, and `std::basic_string< char >::insert()`.

```
5.624.3.67 template<typename _CharT, typename _Traits, typename _Alloc> template<class _InputIterator > void
std::basic_string< _CharT, _Traits, _Alloc >::insert ( iterator __p, _InputIterator __beg, _InputIterator __end )
[inline]
```

Insert a range of characters.

Parameters

<code>__p</code>	Iterator referencing location in string to insert at.
<code>__beg</code>	Start of range.
<code>__end</code>	End of range.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts characters in range `[__beg, __end)`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 3749 of file `basic_string.h`.

```
5.624.3.68 template<typename _CharT, typename _Traits, typename _Alloc> void std::basic_string< _CharT, _Traits, _Alloc
>::insert ( iterator __p, initializer_list< _CharT > __l ) [inline]
```

Insert an `initializer_list` of characters.

Parameters

<code>__p</code>	Iterator referencing location in string to insert at.
<code>__l</code>	The initializer_list of characters to insert.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Definition at line 3760 of file `basic_string.h`.

```
5.624.3.69 template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string<_CharT,
    _Traits, _Alloc>::insert ( size_type __pos1, const basic_string<_CharT, _Traits, _Alloc> &__str ) [inline]
```

Insert value of a string.

Parameters

<code>__pos1</code>	Iterator referencing location in string to insert at.
<code>__str</code>	The string to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts value of `__str` starting at `__pos1`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 3780 of file `basic_string.h`.

```
5.624.3.70 template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string<_CharT,
    _Traits, _Alloc>::insert ( size_type __pos1, const basic_string<_CharT, _Traits, _Alloc> &__str, size_type __pos2,
    size_type __n ) [inline]
```

Insert a substring.

Parameters

<code>__pos1</code>	Iterator referencing location in string to insert at.
<code>__str</code>	The string to insert.
<code>__pos2</code>	Start of characters in <code>str</code> to insert.
<code>__n</code>	Number of characters to insert.

Returns

Reference to this string.

Exceptions

<i>std::length_error</i>	If new length exceeds <code>max_size()</code> .
<i>std::out_of_range</i>	If <code>pos1 > size()</code> or <code>__pos2 > str.size()</code> .

Starting at `pos1`, insert `__n` character of `__str` beginning with `__pos2`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. If `__pos1` is beyond the end of this string or `__pos2` is beyond the end of `__str`, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 3802 of file `basic_string.h`.

5.624.3.71 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string< _CharT, _Traits, _Alloc > & std::basic_string< _CharT, _Traits, _Alloc >::insert (size_type __pos, const _CharT * __s, size_type __n)`

Insert a C substring.

Parameters

<code>__pos</code>	Iterator referencing location in string to insert at.
<code>__s</code>	The C string to insert.
<code>__n</code>	The number of characters to insert.

Returns

Reference to this string.

Exceptions

<i>std::length_error</i>	If new length exceeds <code>max_size()</code> .
<i>std::out_of_range</i>	If <code>__pos</code> is beyond the end of this string.

Inserts the first `__n` characters of `__s` starting at `__pos`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. If `__pos` is beyond `end()`, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 800 of file `basic_string.tcc`.

References `std::basic_string< _CharT, _Traits, _Alloc >::erase()`.

5.624.3.72 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string< _CharT, _Traits, _Alloc >::insert (size_type __pos, const _CharT * __s) [inline]`

Insert a C string.

Parameters

<code>__pos</code>	Iterator referencing location in string to insert at.
<code>__s</code>	The C string to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
<code>std::out_of_range</code>	If <code>pos</code> is beyond the end of this string.

Inserts the first *n* characters of `__s` starting at `__pos`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. If `__pos` is beyond `end()`, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 3843 of file `basic_string.h`.

5.624.3.73 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string<_CharT, _Traits, _Alloc>::insert (size_type __pos, size_type __n, _CharT __c) [inline]`

Insert multiple characters.

Parameters

<code>__pos</code>	Index in string to insert at.
<code>__n</code>	Number of characters to insert
<code>__c</code>	The character to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
<code>std::out_of_range</code>	If <code>__pos</code> is beyond the end of this string.

Inserts `__n` copies of character `__c` starting at index `__pos`. If adding characters causes the length to exceed `max_size()`, `length_error` is thrown. If `__pos > length()`, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 3866 of file `basic_string.h`.

5.624.3.74 `template<typename _CharT, typename _Traits, typename _Alloc> iterator std::basic_string<_CharT, _Traits, _Alloc>::insert(iterator __p, _CharT __c) [inline]`

Insert one character.

Parameters

<code>__p</code>	Iterator referencing position in string to insert at.
<code>__c</code>	The character to insert.

Returns

Iterator referencing newly inserted char.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Inserts character `__c` at position referenced by `__p`. If adding character causes the length to exceed `max_size()`, `length_error` is thrown. If `__p` is beyond end of string, `out_of_range` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 3884 of file `basic_string.h`.

5.624.3.75 `template<typename _CharT, typename _Traits, typename _Alloc> size_type std::basic_string<_CharT, _Traits, _Alloc>::length() const [inline], [noexcept]`

Returns the number of characters in the string, not including any null-termination.

Definition at line 3245 of file `basic_string.h`.

Referenced by `__gnu_profile::__report()`, `std::locale::combine()`, `std::collate<_CharT>::do_compare()`, and `std::collate<_CharT>::do_transform()`.

5.624.3.76 `template<typename _CharT, typename _Traits, typename _Alloc> size_type std::basic_string<_CharT, _Traits, _Alloc>::max_size() const [inline], [noexcept]`

Returns the `size()` of the largest possible string.

Definition at line 3250 of file `basic_string.h`.

Referenced by `std::getline()`, and `std::operator>>()`.

5.624.3.77 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string<_CharT, _Traits, _Alloc>::operator+=(const basic_string<_CharT, _Traits, _Alloc> &__str) [inline]`

Append a string to this string.

Parameters

<code>__str</code>	The string to append.
--------------------	-----------------------

Returns

Reference to this string.

Definition at line 3478 of file `basic_string.h`.

```
5.624.3.78  template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string< _CharT,
           _Traits, _Alloc >::operator+=( const _CharT * __s )  [inline]
```

Append a C string.

Parameters

<code>__c</code>	The C string to append.
------------------	-------------------------

Returns

Reference to this string.

Definition at line 3487 of file `basic_string.h`.

```
5.624.3.79  template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string< _CharT,
           _Traits, _Alloc >::operator+=( _CharT __c )  [inline]
```

Append a character.

Parameters

<code>__c</code>	The character to append.
------------------	--------------------------

Returns

Reference to this string.

Definition at line 3496 of file `basic_string.h`.

```
5.624.3.80  template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string< _CharT,
           _Traits, _Alloc >::operator+=( initializer_list< _CharT > __l )  [inline]
```

Append an `initializer_list` of characters.

Parameters

<code>↵</code>	The initializer_list of characters to be appended.
<code>_↵</code>	
<code>↵</code>	
<code>_↵</code>	
<code>/</code>	

Returns

Reference to this string.

Definition at line 3509 of file basic_string.h.

5.624.3.81 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string< _CharT, _Traits, _Alloc >::operator= (const basic_string< _CharT, _Traits, _Alloc > &__str) [inline]`

Assign the value of *str* to this string.

Parameters

<code>__str</code>	Source string.
--------------------	----------------

Definition at line 3069 of file basic_string.h.

5.624.3.82 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string< _CharT, _Traits, _Alloc >::operator= (const _CharT *__s) [inline]`

Copy contents of *s* into this string.

Parameters

<code>_↵</code>	Source null-terminated string.
<code>_s</code>	

Definition at line 3077 of file basic_string.h.

5.624.3.83 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string< _CharT, _Traits, _Alloc >::operator= (_CharT __c) [inline]`

Set value to string of length 1.

Parameters

<code>_↵</code>	Source character.
<code>_c</code>	

Assigning to a character makes this string length 1 and `(*this)[0] == c`.

Definition at line 3088 of file `basic_string.h`.

5.624.3.84 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string< _CharT, _Traits, _Alloc >::operator= (basic_string< _CharT, _Traits, _Alloc > && __str) [inline]`

Move assign the value of *str* to this string.

Parameters

<code>__str</code>	Source string.
--------------------	----------------

The contents of *str* are moved into this string (without copying). *str* is a valid, but unspecified string.

Definition at line 3104 of file `basic_string.h`.

5.624.3.85 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string< _CharT, _Traits, _Alloc >::operator= (initializer_list< _CharT > __l) [inline]`

Set value to string constructed from initializer list.

Parameters

<code>↵</code>	<code>std::initializer_list.</code>
<code>__↵</code>	
<code>↵</code>	
<code>__↵</code>	
<code>l</code>	

Definition at line 3116 of file `basic_string.h`.

5.624.3.86 `template<typename _CharT, typename _Traits, typename _Alloc> const_reference std::basic_string< _CharT, _Traits, _Alloc >::operator[] (size_type __pos) const [inline], [noexcept]`

Subscript access to the data contained in the string.

Parameters

<code>__pos</code>	The index of the character to access.
--------------------	---------------------------------------

Returns

Read-only (constant) reference to the character.

This operator allows for easy, array-style, data access. Note that data access with this operator is unchecked and `out_of_range` lookups are not defined. (For checked lookups see `at()`.)

Definition at line 3353 of file basic_string.h.

Referenced by std::basic_string< char >::back(), and std::basic_string< char >::front().

5.624.3.87 `template<typename _CharT, typename _Traits, typename _Alloc> reference std::basic_string< _CharT, _Traits, _Alloc >::operator[](size_type __pos) [inline]`

Subscript access to the data contained in the string.

Parameters

<code>__pos</code>	The index of the character to access.
--------------------	---------------------------------------

Returns

Read/write reference to the character.

This operator allows for easy, array-style, data access. Note that data access with this operator is unchecked and `out_of_range` lookups are not defined. (For checked lookups see `at()`.) Unshares the string.

Definition at line 3370 of file basic_string.h.

5.624.3.88 `template<typename _CharT, typename _Traits, typename _Alloc> void std::basic_string< _CharT, _Traits, _Alloc >::pop_back() [inline]`

Remove the last character.

The string must be non-empty.

Definition at line 3954 of file basic_string.h.

5.624.3.89 `template<typename _CharT, typename _Traits, typename _Alloc> void std::basic_string< _CharT, _Traits, _Alloc >::push_back(_CharT __c) [inline]`

Append a single character.

Parameters

<code>__c</code>	Character to append.
------------------	----------------------

Definition at line 3598 of file basic_string.h.

Referenced by std::collate< _CharT >::do_transform(), std::basic_string< char >::operator+=(), std::tr2::operator>>(), and std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow().

5.624.3.90 `template<typename _CharT, typename _Traits, typename _Alloc> reverse_iterator std::basic_string< _CharT, _Traits, _Alloc >::rbegin () [inline]`

Returns a read/write reverse iterator that points to the last character in the string. Iteration is done in reverse element order. Unshares the string.

Definition at line 3168 of file basic_string.h.

5.624.3.91 `template<typename _CharT, typename _Traits, typename _Alloc> const_reverse_iterator std::basic_string< _CharT, _Traits, _Alloc >::rbegin () const [inline],[noexcept]`

Returns a read-only (constant) reverse iterator that points to the last character in the string. Iteration is done in reverse element order.

Definition at line 3177 of file basic_string.h.

5.624.3.92 `template<typename _CharT, typename _Traits, typename _Alloc> reverse_iterator std::basic_string< _CharT, _Traits, _Alloc >::rend () [inline]`

Returns a read/write reverse iterator that points to one before the first character in the string. Iteration is done in reverse element order. Unshares the string.

Definition at line 3186 of file basic_string.h.

5.624.3.93 `template<typename _CharT, typename _Traits, typename _Alloc> const_reverse_iterator std::basic_string< _CharT, _Traits, _Alloc >::rend () const [inline],[noexcept]`

Returns a read-only (constant) reverse iterator that points to one before the first character in the string. Iteration is done in reverse element order.

Definition at line 3195 of file basic_string.h.

5.624.3.94 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string< _CharT, _Traits, _Alloc >::replace (size_type __pos, size_type __n, const basic_string< _CharT, _Traits, _Alloc > & __str) [inline]`

Replace characters with value from another string.

Parameters

<code>__pos</code>	Index of first character to replace.
<code>__n</code>	Number of characters to be replaced.
<code>__str</code>	String to insert.

Returns

Reference to this string.

Exceptions

<i>std::out_of_range</i>	If <i>pos</i> is beyond the end of this string.
<i>std::length_error</i>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range `[__pos, __pos+__n)` from this string. In place, the value of `__str` is inserted. If `__pos` is beyond end of string, `out_of_range` is thrown. If the length of the result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 3979 of file `basic_string.h`.

Referenced by `std::basic_string< char >::append()`, `std::basic_string< char >::assign()`, `std::basic_string< _CharT, _Traits, _Alloc >::erase()`, `std::basic_string< char >::insert()`, and `std::basic_string< char >::replace()`.

5.624.3.95 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string< _CharT, _Traits, _Alloc >::replace (size_type __pos1, size_type __n1, const basic_string< _CharT, _Traits, _Alloc > & __str, size_type __pos2, size_type __n2) [inline]`

Replace characters with value from another string.

Parameters

<code>__pos1</code>	Index of first character to replace.
<code>__n1</code>	Number of characters to be replaced.
<code>__str</code>	String to insert.
<code>__pos2</code>	Index of first character of <code>str</code> to use.
<code>__n2</code>	Number of characters from <code>str</code> to use.

Returns

Reference to this string.

Exceptions

<i>std::out_of_range</i>	If <code>__pos1 > size()</code> or <code>__pos2 > __str.size()</code> .
<i>std::length_error</i>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range `[__pos1, __pos1 + n)` from this string. In place, the value of `__str` is inserted. If `__pos` is beyond end of string, `out_of_range` is thrown. If the length of the result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 4001 of file `basic_string.h`.

5.624.3.96 `template<typename _CharT, typename _Traits, typename _Alloc > basic_string< _CharT, _Traits, _Alloc > & std::basic_string< _CharT, _Traits, _Alloc >::replace (size_type __pos, size_type __n1, const _CharT * __s, size_type __n2)`

Replace characters with value of a C substring.

Parameters

<code>__pos</code>	Index of first character to replace.
<code>__n1</code>	Number of characters to be replaced.
<code>__s</code>	C string to insert.
<code>__n2</code>	Number of characters from <code>s</code> to use.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <code>pos1 > size()</code> .
<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range `[__pos, __pos + __n1)` from this string. In place, the first `__n2` characters of `__s` are inserted, or all of `__s` if `__n2` is too large. If `__pos` is beyond end of string, `out_of_range` is thrown. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 854 of file `basic_string.tcc`.

References `std::basic_string<_CharT, _Traits, _Alloc>::capacity()`, `std::basic_string<_CharT, _Traits, _Alloc>::get_allocator()`, `std::basic_string<_CharT, _Traits, _Alloc>::reserve()`, and `std::basic_string<_CharT, _Traits, _Alloc>::size()`.

5.624.3.97 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string<_CharT, _Traits, _Alloc>::replace (size_type __pos, size_type __n1, const _CharT * __s) [inline]`

Replace characters with value of a C string.

Parameters

<code>__pos</code>	Index of first character to replace.
<code>__n1</code>	Number of characters to be replaced.
<code>__s</code>	C string to insert.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <code>pos > size()</code> .
<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range [`__pos`, `__pos + __n1`) from this string. In place, the characters of `__s` are inserted. If `__pos` is beyond end of string, `out_of_range` is thrown. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 4046 of file `basic_string.h`.

```
5.624.3.98 template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string<_CharT,
    _Traits, _Alloc>::replace ( size_type __pos, size_type __n1, size_type __n2, _CharT __c ) [inline]
```

Replace characters with multiple characters.

Parameters

<code>__pos</code>	Index of first character to replace.
<code>__n1</code>	Number of characters to be replaced.
<code>__n2</code>	Number of characters to insert.
<code>__c</code>	Character to insert.

Returns

Reference to this string.

Exceptions

<code>std::out_of_range</code>	If <code>__pos > size()</code> .
<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .

Removes the characters in the range [`pos`, `pos + n1`) from this string. In place, `__n2` copies of `__c` are inserted. If `__pos` is beyond end of string, `out_of_range` is thrown. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 4070 of file `basic_string.h`.

```
5.624.3.99 template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string<_CharT,
    _Traits, _Alloc>::replace ( iterator __i1, iterator __i2, const basic_string<_CharT, _Traits, _Alloc> & __str )
    [inline]
```

Replace range of characters with string.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__str</code>	String value to insert.

Returns

Reference to this string.

Exceptions

<i>std::length_error</i>	If new length exceeds <code>max_size()</code> .
--------------------------	---

Removes the characters in the range `[__i1, __i2)`. In place, the value of `__str` is inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 4088 of file `basic_string.h`.

5.624.3.100 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string<_CharT, _Traits, _Alloc>::replace (iterator __i1, iterator __i2, const _CharT *__s, size_type __n) [inline]`

Replace range of characters with C substring.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__s</code>	C string value to insert.
<code>__n</code>	Number of characters from <code>s</code> to insert.

Returns

Reference to this string.

Exceptions

<i>std::length_error</i>	If new length exceeds <code>max_size()</code> .
--------------------------	---

Removes the characters in the range `[__i1, __i2)`. In place, the first `__n` characters of `__s` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 4107 of file `basic_string.h`.

5.624.3.101 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string<_CharT, _Traits, _Alloc>::replace (iterator __i1, iterator __i2, const _CharT *__s) [inline]`

Replace range of characters with C string.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__s</code>	C string value to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[__i1, __i2)`. In place, the characters of `__s` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 4128 of file `basic_string.h`.

5.624.3.102 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string<_CharT, _Traits, _Alloc>::replace (iterator __i1, iterator __i2, size_type __n, _CharT __c) [inline]`

Replace range of characters with multiple characters.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__n</code>	Number of characters to insert.
<code>__c</code>	Character to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[__i1,__i2)`. In place, `__n` copies of `__c` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 4149 of file `basic_string.h`.

5.624.3.103 `template<typename _CharT, typename _Traits, typename _Alloc> template<class _InputIterator > basic_string& std::basic_string< _CharT, _Traits, _Alloc >::replace (iterator __i1, iterator __i2, _InputIterator __k1, _InputIterator __k2) [inline]`

Replace range of characters with range.

Parameters

<code>__i1</code>	Iterator referencing start of range to replace.
<code>__i2</code>	Iterator referencing end of range to replace.
<code>__k1</code>	Iterator referencing start of range to insert.
<code>__k2</code>	Iterator referencing end of range to insert.

Returns

Reference to this string.

Exceptions

<code>std::length_error</code>	If new length exceeds <code>max_size()</code> .
--------------------------------	---

Removes the characters in the range `[__i1,__i2)`. In place, characters in the range `[__k1,__k2)` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 4173 of file `basic_string.h`.

5.624.3.104 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string& std::basic_string< _CharT, _Traits, _Alloc >::replace (iterator __i1, iterator __i2, initializer_list<_CharT> __l) [inline]`

Replace range of characters with `initializer_list`.

Parameters

<code>↔ __i1</code>	Iterator referencing start of range to replace.
<code>↔ __i2</code>	Iterator referencing end of range to replace.
<code>↔ __l</code>	The <code>initializer_list</code> of characters to insert.

Returns

Reference to this string.

Exceptions

<i>std::length_error</i>	If new length exceeds <code>max_size()</code> .
--------------------------	---

Removes the characters in the range `[__i1, __i2)`. In place, characters in the range `[__k1, __k2)` are inserted. If the length of result exceeds `max_size()`, `length_error` is thrown. The value of the string doesn't change if an error is thrown.

Definition at line 4242 of file `basic_string.h`.

5.624.3.105 `template<typename _CharT, typename _Traits, typename _Alloc > void std::basic_string< _CharT, _Traits, _Alloc >::reserve (size_type __res_arg = 0)`

Attempt to preallocate enough memory for specified number of characters.

Parameters

<code>__res_arg</code>	Number of characters required.
------------------------	--------------------------------

Exceptions

<i>std::length_error</i>	If <code>__res_arg</code> exceeds <code>max_size()</code> .
--------------------------	---

This function attempts to reserve enough memory for the string to hold the specified number of characters. If the number requested is more than `max_size()`, `length_error` is thrown.

The advantage of this function is that if optimal code is a necessity and the user can determine the string length that will be required, the user can reserve the memory in advance, and thus prevent a possible reallocation of memory and copying of string data.

Definition at line 942 of file `basic_string.tcc`.

References `std::basic_string< _CharT, _Traits, _Alloc >::capacity()`, `std::basic_string< _CharT, _Traits, _Alloc >::get_allocator()`, `std::basic_string< _CharT, _Traits, _Alloc >::size()`, and `std::basic_string< _CharT, _Traits, _Alloc >::swap()`.

Referenced by `std::basic_string< _CharT, _Traits, _Alloc >::append()`, `std::basic_string< char >::capacity()`, `std::basic_string< _CharT, _Traits, _Alloc >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::tr2::operator>>()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow()`, `std::basic_string< char >::push_back()`, `std::basic_string< _CharT, _Traits, _Alloc >::replace()`, and `std::basic_string< char >::shrink_to_fit()`.

5.624.3.106 `template<typename _CharT, typename _Traits, typename _Alloc > void std::basic_string< _CharT, _Traits, _Alloc >::resize (size_type __n, _CharT __c)`

Resizes the string to the specified number of characters.

Parameters

<code>__n</code>	Number of characters the string should contain.
<code>__c</code>	Character to fill any new elements.

This function will resize the string to the specified number of characters. If the number is smaller than the string's current size the string is truncated, otherwise the string is extended and new elements are set to `__c`.

Definition at line 1080 of file `basic_string.tcc`.

References `std::basic_string<_CharT, _Traits, _Alloc>::append()`, `std::basic_string<_CharT, _Traits, _Alloc>::copy()`, `std::basic_string<_CharT, _Traits, _Alloc>::erase()`, and `std::basic_string<_CharT, _Traits, _Alloc>::size()`.

Referenced by `std::money_get<_CharT, _InIter>::do_get()`, `std::basic_string<char>::max_size()`, `std::basic_string<char>::resize()`, and `std::basic_string<_CharT, _Traits, _Alloc>::swap()`.

5.624.3.107 `template<typename _CharT, typename _Traits, typename _Alloc> void std::basic_string<_CharT, _Traits, _Alloc>::resize(size_type __n) [inline]`

Resizes the string to the specified number of characters.

Parameters

<code>__n</code>	Number of characters the string should contain.
------------------	---

This function will resize the string to the specified length. If the new size is smaller than the string's current size the string is truncated, otherwise the string is extended and new characters are default-constructed. For basic types such as `char`, this means setting them to 0.

Definition at line 3277 of file `basic_string.h`.

5.624.3.108 `template<typename _CharT, typename _Traits, typename _Alloc> size_type std::basic_string<_CharT, _Traits, _Alloc>::rfind(const basic_string<_CharT, _Traits, _Alloc> & __str, size_type __pos = npos) const [inline], [noexcept]`

Find last position of a string.

Parameters

<code>__str</code>	String to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of start of last occurrence.

Starting from `__pos`, searches backward for value of `__str` within this string. If found, returns the index where it begins. If not found, returns `npos`.

Definition at line 4442 of file `basic_string.h`.

Referenced by `std::basic_string< _CharT, _Traits, _Alloc >::find()`, `std::basic_string< char >::find_last_of()`, `std::basic_string< char >::rfind()`, and `std::basic_string< _CharT, _Traits, _Alloc >::rfind()`.

5.624.3.109 `template<typename _CharT, typename _Traits, typename _Alloc > basic_string< _CharT, _Traits, _Alloc >::size_type std::basic_string< _CharT, _Traits, _Alloc >::rfind (const _CharT * __s, size_type __pos, size_type __n) const`

Find last position of a C substring.

Parameters

<code>__s</code>	C string to locate.
<code>__pos</code>	Index of character to search back from.
<code>__n</code>	Number of characters from <code>s</code> to search for.

Returns

Index of start of last occurrence.

Starting from `__pos`, searches backward for the first `__n` characters in `__s` within this string. If found, returns the index where it begins. If not found, returns `npos`.

Definition at line 1219 of file `basic_string.tcc`.

References `std::min()`, `std::basic_string< _CharT, _Traits, _Alloc >::npos`, `std::basic_string< _CharT, _Traits, _Alloc >::rfind()`, and `std::basic_string< _CharT, _Traits, _Alloc >::size()`.

5.624.3.110 `template<typename _CharT, typename _Traits, typename _Alloc> size_type std::basic_string< _CharT, _Traits, _Alloc >::rfind (const _CharT * __s, size_type __pos = npos) const [inline]`

Find last position of a C string.

Parameters

<code>__s</code>	C string to locate.
<code>__pos</code>	Index of character to start search at (default end).

Returns

Index of start of last occurrence.

Starting from `__pos`, searches backward for the value of `__s` within this string. If found, returns the index where it begins. If not found, returns `npos`.

Definition at line 4472 of file `basic_string.h`.

5.624.3.111 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string< _CharT, _Traits, _Alloc>::size_type std::basic_string< _CharT, _Traits, _Alloc>::rfind (_CharT __c, size_type __pos = npos) const [noexcept]`

Find last position of a character.

Parameters

<code>__c</code>	Character to locate.
<code>__pos</code>	Index of character to search back from (default end).

Returns

Index of last occurrence.

Starting from `__pos`, searches backward for `__c` within this string. If found, returns the index where it was found. If not found, returns `npos`.

Definition at line 1240 of file `basic_string.tcc`.

References `std::basic_string< _CharT, _Traits, _Alloc>::find_first_of()`, `std::basic_string< _CharT, _Traits, _Alloc>::npos`, and `std::basic_string< _CharT, _Traits, _Alloc>::size()`.

5.624.3.112 `template<typename _CharT, typename _Traits, typename _Alloc> void std::basic_string< _CharT, _Traits, _Alloc>::shrink_to_fit () [inline], [noexcept]`

A non-binding request to reduce capacity() to size().

Definition at line 3283 of file `basic_string.h`.

5.624.3.113 `template<typename _CharT, typename _Traits, typename _Alloc> size_type std::basic_string< _CharT, _Traits, _Alloc>::size () const [inline], [noexcept]`

Returns the number of characters in the string, not including any null-termination.

Definition at line 3239 of file `basic_string.h`.

Referenced by `std::basic_string< _CharT, _Traits, _Alloc>::append()`, `std::basic_regex< _Ch_type, _Rx_traits>::assign()`, `std::basic_string< _CharT, _Traits, _Alloc>::assign()`, `std::basic_string< char>::assign()`, `std::basic_string< char>::at()`, `std::basic_string< char>::back()`, `std::basic_string< char>::cend()`, `std::basic_string< char>::clear()`, `std::basic_string< char>::compare()`, `std::basic_string< _CharT, _Traits, _Alloc>::compare()`, `std::money_get< _CharT, _InIter>::do_get()`, `std::basic_string< char>::empty()`, `std::basic_string< char>::end()`, `std::basic_string< _CharT, _Traits, _Alloc>::find()`, `std::basic_string< _CharT, _Traits, _Alloc>::find_first_not_of()`, `std::basic_string< _CharT, _Traits, _Alloc>::find_first_of()`, `std::basic_string< _CharT, _Traits, _Alloc>::find_last_not_of()`, `std::basic_string< _CharT, _Traits, _Alloc>::find_last_of()`, `std::match_results< _Bi_iter>::format()`, `std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc>::from_bytes()`, `std::operator+()`, `std::operator<()`, `std::operator==()`, `std::tr2::operator>>()`, `std::basic_string< char>::operator[]()`, `std::basic_string< char>::pop_back()`, `std::basic_string< char>::push_back()`, `std::basic_string< _CharT, _Traits, _Alloc>::replace()`, `std::basic_string< _CharT, _Traits, _Alloc>::reserve()`, `std::basic_string< _CharT, _Traits, _Alloc>::resize()`, `std::basic_string< _CharT, _Traits, _Alloc>::rfind()`, `std::basic_string< char>::shrink_to_fit()`, `std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc>::to_bytes()`, `std::regex_traits< _Ch_type>::transform()`, and `std::experimental::filesystem::v1::u8path()`.

5.624.3.114 `template<typename _CharT, typename _Traits, typename _Alloc> basic_string std::basic_string< _CharT, _Traits, _Alloc>::substr (size_type __pos = 0, size_type __n = npos) const [inline]`

Get a substring.

Parameters

<code>__pos</code>	Index of first character (default 0).
<code>__n</code>	Number of characters in substring (default remainder).

Returns

The new string.

Exceptions

<code>std::out_of_range</code>	If <code>__pos > size()</code> .
--------------------------------	-------------------------------------

Construct and return a new string using the `__n` characters starting at `__pos`. If the string is too short, use the remainder of the characters. If `__pos` is beyond the end of the string, `out_of_range` is thrown.

Definition at line 4757 of file `basic_string.h`.

Referenced by `__gnu_profile::__report()`.

5.624.3.115 `template<typename _CharT, typename _Traits, typename _Alloc > void std::basic_string< _CharT, _Traits, _Alloc >::swap (basic_string< _CharT, _Traits, _Alloc > & __s)`

Swap contents with another string.

Parameters

<code>__s</code>	String to swap with.
------------------	----------------------

Exchanges the contents of this string with that of `__s` in constant time.

Definition at line 959 of file `basic_string.tcc`.

References `std::basic_string< _CharT, _Traits, _Alloc >::get_allocator()`, and `std::basic_string< _CharT, _Traits, _Alloc >::resize()`.

Referenced by `std::basic_string< char >::assign()`, `std::basic_string< char >::operator=()`, `std::basic_string< char >::replace()`, and `std::basic_string< _CharT, _Traits, _Alloc >::reserve()`.

5.624.4 Member Data Documentation

5.624.4.1 `template<typename _CharT, typename _Traits, typename _Alloc> const basic_string< _CharT, _Traits, _Alloc >::size_type std::basic_string< _CharT, _Traits, _Alloc >::npos [static]`

Value returned by various member functions when they fail.

Definition at line 2801 of file basic_string.h.

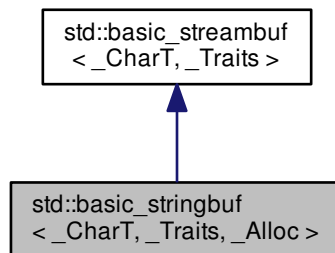
Referenced by `std::basic_string<_CharT, _Traits, _Alloc>::find()`, `std::basic_string<_CharT, _Traits, _Alloc>::find↵_first_not_of()`, `std::basic_string<_CharT, _Traits, _Alloc>::find_first_of()`, `std::basic_string<_CharT, _Traits, _Alloc>::find_last_not_of()`, `std::basic_string<_CharT, _Traits, _Alloc>::find_last_of()`, and `std::basic_string<_CharT, _↵Traits, _Alloc>::rfind()`.

The documentation for this class was generated from the following files:

- [basic_string.h](#)
- [basic_string.tcc](#)

5.625 std::basic_stringbuf<_CharT, _Traits, _Alloc> Class Template Reference

Inheritance diagram for `std::basic_stringbuf<_CharT, _Traits, _Alloc>`:



Public Types

- typedef `__string_type::size_type` **__size_type**
- typedef `basic_streambuf<char_type, traits_type>` **__streambuf_type**
- typedef `basic_string<char_type, _Traits, _Alloc>` **__string_type**
- typedef `_Alloc` **allocator_type**
- typedef `_CharT` **char_type**
- typedef `traits_type::int_type` **int_type**
- typedef `traits_type::off_type` **off_type**
- typedef `traits_type::pos_type` **pos_type**
- typedef `_Traits` **traits_type**

Public Member Functions

- [basic_stringbuf](#) ([ios_base::openmode](#) __mode=[ios_base::in](#)|[ios_base::out](#))
- [basic_stringbuf](#) (const [__string_type](#) &__str, [ios_base::openmode](#) __mode=[ios_base::in](#)|[ios_base::out](#))
- **basic_stringbuf** (const [basic_stringbuf](#) &)=delete
- **basic_stringbuf** ([basic_stringbuf](#) &&__rhs)
- [locale](#) [getloc](#) () const
- [streamsize](#) [in_avail](#) ()
- [basic_stringbuf](#) & **operator=** (const [basic_stringbuf](#) &)=delete
- [basic_stringbuf](#) & **operator=** ([basic_stringbuf](#) &&__rhs)
- [locale](#) [pubimbue](#) (const [locale](#) &__loc)
- int_type [sbumpc](#) ()
- int_type [sgetc](#) ()
- [streamsize](#) [sgetn](#) (char_type *__s, [streamsize](#) __n)
- int_type [snextc](#) ()
- int_type [sputbackc](#) (char_type __c)
- int_type [sputc](#) (char_type __c)
- [streamsize](#) [sputn](#) (const char_type *__s, [streamsize](#) __n)
- [__string_type](#) [str](#) () const
- void [str](#) (const [__string_type](#) &__s)
- int_type [sungetc](#) ()
- void **swap** ([basic_stringbuf](#) &__rhs)
- [basic_streambuf](#) * [pubsetbuf](#) (char_type *__s, [streamsize](#) __n)
- pos_type [pubseekoff](#) (off_type __off, [ios_base::seekdir](#) __way, [ios_base::openmode](#) __mode=[ios_base::in](#)|[ios_base::out](#))
- pos_type [pubseekpos](#) (pos_type __sp, [ios_base::openmode](#) __mode=[ios_base::in](#)|[ios_base::out](#))
- int [pubsync](#) ()

Protected Member Functions

- void [__safe_gbump](#) ([streamsize](#) __n)
- void [__safe_pbump](#) ([streamsize](#) __n)
- void [_M_pbump](#) (char_type *__pbeg, char_type *__pend, off_type __off)
- void [_M_stringbuf_init](#) ([ios_base::openmode](#) __mode)
- void [_M_sync](#) (char_type *__base, __size_type __i, __size_type __o)
- void [_M_update_egptr](#) ()
- void [gbump](#) (int __n)
- virtual void [imbue](#) (const [locale](#) &__loc)
- virtual int_type [overflow](#) (int_type __c=traits_type::eof())
- virtual int_type [pbackfail](#) (int_type __c=traits_type::eof())
- void [pbump](#) (int __n)
- virtual pos_type [seekoff](#) (off_type __off, [ios_base::seekdir](#) __way, [ios_base::openmode](#) __mode=[ios_base::in](#)|[ios_base::out](#))
- virtual pos_type [seekpos](#) (pos_type __sp, [ios_base::openmode](#) __mode=[ios_base::in](#)|[ios_base::out](#))
- virtual [__streambuf_type](#) * [setbuf](#) (char_type *__s, [streamsize](#) __n)
- void [setg](#) (char_type *__gbeg, char_type *__gnext, char_type *__gend)
- void [setp](#) (char_type *__pbeg, char_type *__pend)
- virtual [streamsize](#) [showmanyc](#) ()
- void **swap** ([basic_streambuf](#) &__sb)

- virtual int [sync](#) ()
- virtual int_type [uflow](#) ()
- virtual int_type [underflow](#) ()
- virtual [streamsize](#) [xsgetn](#) (char_type *__s, [streamsize](#) __n)
- virtual [streamsize](#) [xspn](#) (const char_type *__s, [streamsize](#) __n)
- char_type * [eback](#) () const
- char_type * [gptr](#) () const
- char_type * [egptr](#) () const
- char_type * [pbase](#) () const
- char_type * [pptr](#) () const
- char_type * [epptr](#) () const

Protected Attributes

- [locale](#) [_M_buf_locale](#)
- char_type * [_M_in_beg](#)
- char_type * [_M_in_cur](#)
- char_type * [_M_in_end](#)
- [ios_base::openmode](#) [_M_mode](#)
- char_type * [_M_out_beg](#)
- char_type * [_M_out_cur](#)
- char_type * [_M_out_end](#)
- [__string_type](#) [_M_string](#)

5.625.1 Detailed Description

```
template<typename _CharT, typename _Traits, typename _Alloc>
class std::basic_stringbuf< _CharT, _Traits, _Alloc >
```

The actual work of input and output (for `std::string`).

Template Parameters

<code>_CharT</code>	Type of character stream.
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_CharT></code> .

This class associates either or both of its input and output sequences with a sequence of characters, which can be initialized from, or made available as, a `std::basic_string`. (Paraphrased from [27.7.1]/1.)

For this class, open modes (of type `ios_base::openmode`) have `in` set if the input sequence can be read, and `out` set if the output sequence can be written.

Definition at line 96 of file `iosfwd`.

5.625.2 Constructor & Destructor Documentation

5.625.2.1 `template<typename _CharT, typename _Traits, typename _Alloc> std::basic_stringbuf< _CharT, _Traits, _Alloc >::basic_stringbuf (ios_base::openmode __mode = ios_base::in | ios_base::out) [inline], [explicit]`

Starts with an empty string buffer.

Parameters

<code>__mode</code>	Whether the buffer can read, or write, or both.
---------------------	---

The default constructor initializes the parent class using its own default ctor.

Definition at line 100 of file sstream.

5.625.2.2 `template<typename _CharT, typename _Traits, typename _Alloc> std::basic_stringbuf< _CharT, _Traits, _Alloc >::basic_stringbuf (const __string_type & __str, ios_base::openmode __mode = ios_base::in | ios_base::out) [inline], [explicit]`

Starts with an existing string buffer.

Parameters

<code>__str</code>	A string to copy as a starting buffer.
<code>__mode</code>	Whether the buffer can read, or write, or both.

This constructor initializes the parent class using its own default ctor.

Definition at line 113 of file sstream.

5.625.3 Member Function Documentation

5.625.3.1 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::eback () const [inline], [protected], [inherited]`

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Definition at line 482 of file streambuf.

Referenced by `std::basic_filebuf< _CharT, _Traits >::imbue()`, `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, `std::basic_filebuf< _CharT, _Traits >::seekpos()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

5.625.3.2 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::egptr ()`
`const [inline], [protected], [inherited]`

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Definition at line 488 of file `streambuf`.

Referenced by `std::ostreambuf_iterator< _CharT, _Traits >::failed()`, `std::basic_filebuf< _CharT, _Traits >::seekpos()`, `std::basic_filebuf< _CharT, _Traits >::showmanyc()`, `std::basic_filebuf< _CharT, _Traits >::underflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsgetn()`.

5.625.3.3 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::eptr ()`
`const [inline], [protected], [inherited]`

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

Definition at line 535 of file `streambuf`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::xspn()`.

5.625.3.4 `template<typename _CharT, typename _Traits> void std::basic_streambuf< _CharT, _Traits >::gbump (int __n)`
`[inline], [protected], [inherited]`

Moving the read position.

Parameters

<code>__n</code>	The delta by which to move.
------------------	-----------------------------

This just advances the read position without returning any data.

Definition at line 498 of file `streambuf`.

Referenced by std::basic_filebuf< _CharT, _Traits >::pbackfail(), and std::basic_filebuf< _CharT, _Traits >::xsgetn().

5.625.3.5 template<typename _CharT, typename _Traits> locale std::basic_streambuf< _CharT, _Traits >::getloc () const
[inline], [inherited]

Locale access.

Returns

The current locale in effect.

If pubimbue(loc) has been called, then the most recent loc is returned. Otherwise the global locale in effect at the time of construction is returned.

Definition at line 226 of file streambuf.

5.625.3.6 template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::gptr ()
const [inline], [protected], [inherited]

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- eback() returns the beginning pointer for the input sequence
- gptr() returns the next pointer for the input sequence
- egptr() returns the end pointer for the input sequence

Definition at line 485 of file streambuf.

Referenced by std::ostreambuf_iterator< _CharT, _Traits >::failed(), std::basic_filebuf< _CharT, _Traits >::imbue(), std::basic_filebuf< _CharT, _Traits >::pbackfail(), std::basic_filebuf< _CharT, _Traits >::seekpos(), std::basic_filebuf< _CharT, _Traits >::showmanyc(), std::basic_filebuf< _CharT, _Traits >::underflow(), and std::basic_filebuf< _CharT, _Traits >::xsgetn().

5.625.3.7 template<typename _CharT, typename _Traits> virtual void std::basic_streambuf< _CharT, _Traits >::imbue (const locale & __loc) [inline], [protected], [virtual], [inherited]

Changes translations.

Parameters

<code>__loc</code>	A new locale.
--------------------	---------------

Translations done during I/O which depend on the current locale are changed by this call. The standard adds, *Between invocations of this function a class derived from streambuf can safely cache results of calls to locale functions and to members of facets so obtained.*

Note

Base class version does nothing.

Reimplemented in `std::basic_filebuf<_CharT, _Traits>`, `std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>`, and `std::basic_filebuf<char_type, traits_type>`.

Definition at line 576 of file `streambuf`.

5.625.3.8 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf<_CharT, _Traits>::in_avail ()`
`[inline], [inherited]`

Looking ahead into the stream.

Returns

The number of characters available.

If a read position is available, returns the number of characters available for reading before the buffer must be refilled. Otherwise returns the derived `showmanyc()`.

Definition at line 284 of file `streambuf`.

5.625.3.9 `template<class _CharT, class _Traits, class _Alloc> basic_stringbuf<_CharT, _Traits, _Alloc>::int_type`
`std::basic_stringbuf<_CharT, _Traits, _Alloc>::overflow (int_type __c = traits_type::eof())`
`[protected], [virtual]`

Consumes data from the buffer; writes to the controlled sequence.

Parameters

<code>__c</code>	An additional character to consume.
<code>__c</code>	

Returns

`eof()` to indicate failure, something else (usually `__c`, or `not_eof()`)

Informally, this function is called when the output buffer is full (or does not exist, as buffering need not actually be done). If a buffer exists, it is *consumed*, with *some effect* on the controlled sequence. (Typically, the buffer is written out to the sequence verbatim.) In either case, the character `c` is also written out, if `__c` is not `eof()`.

For a formal definition of this function, see a good text such as Langer & Kreft, or [27.5.2.4.5]/3-7.

A functioning output `streambuf` can be created by overriding only this function (no buffer area will be used).

Note

Base class version does nothing, returns eof().

Reimplemented from [std::basic_streambuf< _CharT, _Traits >](#).

Definition at line 80 of file sstream.tcc.

References `std::basic_string< _CharT, _Traits, _Alloc >::assign()`, `std::max()`, `std::min()`, `std::ios_base::out`, `std::basic_<_>string< _CharT, _Traits, _Alloc >::push_back()`, `std::basic_string< _CharT, _Traits, _Alloc >::reserve()`, and `std::<_>basic_stringbuf< _CharT, _Traits, _Alloc >::underflow()`.

Referenced by `std::basic_stringbuf< _CharT, _Traits, _Alloc >::pbackfail()`.

```
5.625.3.10 template<class _CharT, class _Traits, class _Alloc> basic_stringbuf< _CharT, _Traits, _Alloc >::int_type
std::basic_stringbuf< _CharT, _Traits, _Alloc >::pbackfail ( int_type __c = traits_type::eof() )
[protected], [virtual]
```

Tries to back up the input sequence.

Parameters

<code><_></code>	The character to be inserted back into the sequence.
<code>_C</code>	

Returns

eof() on failure, *some other value* on success

Postcondition

The constraints of `gptr()`, `eback()`, and `pptr()` are the same as for `underflow()`.

Note

Base class version does nothing, returns eof().

Reimplemented from [std::basic_streambuf< _CharT, _Traits >](#).

Definition at line 46 of file sstream.tcc.

References `std::ios_base::out`, and `std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow()`.

5.625.3.11 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::pbase ()`
`const [inline], [protected], [inherited]`

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

Definition at line 529 of file `streambuf`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_filebuf< _CharT, _Traits >::seekoff()`, `std::basic_filebuf< _CharT, _Traits >::seekpos()`, `std::basic_filebuf< _CharT, _Traits >::sync()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

5.625.3.12 `template<typename _CharT, typename _Traits> void std::basic_streambuf< _CharT, _Traits >::pbump (int __n)`
`[inline], [protected], [inherited]`

Moving the write position.

Parameters

<code>__n</code>	The delta by which to move.
------------------	-----------------------------

This just advances the write position without returning any data.

Definition at line 545 of file `streambuf`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`.

5.625.3.13 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::pptr ()`
`const [inline], [protected], [inherited]`

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

Definition at line 532 of file `streambuf`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_filebuf< _CharT, _Traits >::seekoff()`, `std::basic_filebuf< _CharT, _Traits >::seekpos()`, `std::basic_filebuf< _CharT, _Traits >::sync()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

5.625.3.14 `template<typename _CharT, typename _Traits> locale std::basic_streambuf< _CharT, _Traits >::pubimbue (const locale & __loc) [inline], [inherited]`

Entry point for imbue().

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls the derived imbue(__loc).

Definition at line 209 of file streambuf.

5.625.3.15 `template<typename _CharT, typename _Traits> pos_type std::basic_streambuf< _CharT, _Traits >::pubseekoff (off_type __off, ios_base::seekdir __way, ios_base::openmode __mode = ios_base::in | ios_base::out) [inline], [inherited]`

Alters the stream position.

Parameters

<code>__off</code>	Offset.
<code>__way</code>	Value for ios_base::seekdir.
<code>__mode</code>	Value for ios_base::openmode.

Calls virtual seekoff function.

Definition at line 251 of file streambuf.

5.625.3.16 `template<typename _CharT, typename _Traits> pos_type std::basic_streambuf< _CharT, _Traits >::pubseekpos (pos_type __sp, ios_base::openmode __mode = ios_base::in | ios_base::out) [inline], [inherited]`

Alters the stream position.

Parameters

<code>__sp</code>	Position
<code>__mode</code>	Value for ios_base::openmode.

Calls virtual seekpos function.

Definition at line 263 of file streambuf.

5.625.3.17 `template<typename _CharT, typename _Traits> basic_streambuf* std::basic_streambuf<_CharT, _Traits>::pubsetbuf(char_type* __s, streamsize __n) [inline], [inherited]`

Entry points for derived buffer functions.

The public versions of `pubfoo` dispatch to the protected derived `foo` member functions, passing the arguments (if any) and returning the result unchanged.

Definition at line 239 of file `streambuf`.

5.625.3.18 `template<typename _CharT, typename _Traits> int std::basic_streambuf<_CharT, _Traits>::pubsync() [inline], [inherited]`

Calls virtual `sync` function.

Definition at line 271 of file `streambuf`.

Referenced by `std::basic_istream<_CharT, _Traits>::sync()`.

5.625.3.19 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::sbumpc() [inline], [inherited]`

Getting the next character.

Returns

The next character, or `eof`.

If the input read position is available, returns that character and increments the read pointer, otherwise calls and returns `uflow()`.

Definition at line 316 of file `streambuf`.

Referenced by `std::basic_istream<_CharT, _Traits>::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, and `std::istreambuf_iterator<_CharT, _Traits>::operator++()`.

5.625.3.20 `template<class _CharT, class _Traits, class _Alloc> basic_stringbuf<_CharT, _Traits, _Alloc>::pos_type std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekoff(off_type, ios_base::seekdir, ios_base::openmode = ios_base::in | ios_base::out) [protected], [virtual]`

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented from [std::basic_streambuf<_CharT, _Traits>](#).

Definition at line 149 of file `sstream.tcc`.

References `std::ios_base::cur`, `std::ios_base::end`, `std::ios_base::in`, `std::ios_base::out`, and `std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekpos()`.

Referenced by `std::basic_stringbuf<_CharT, _Traits, _Alloc>::underflow()`.

```
5.625.3.21 template<class _CharT, class _Traits, class _Alloc > basic_stringbuf< _CharT, _Traits, _Alloc >::pos_type
std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos ( pos_type , ios_base::openmode =
ios_base::in | ios_base::out ) [protected], [virtual]
```

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Reimplemented from [std::basic_streambuf< _CharT, _Traits >](#).

Definition at line 197 of file `sstream.tcc`.

References `std::ios_base::in`, and `std::ios_base::out`.

Referenced by `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`.

```
5.625.3.22 template<typename _CharT, typename _Traits, typename _Alloc> virtual __streambuf_type*
std::basic_stringbuf< _CharT, _Traits, _Alloc >::setbuf ( char_type * __s, streamsize __n ) [inline],
[protected], [virtual]
```

Manipulates the buffer.

Parameters

<code>__s</code>	Pointer to a buffer area.
<code>__n</code>	Size of <code>__s</code> .

Returns

`this`

If no buffer has already been created, and both `__s` and `__n` are non-zero, then `__s` is used as a buffer; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html#io.streambuf.buffering> for more.

Reimplemented from [std::basic_streambuf< _CharT, _Traits >](#).

Definition at line 243 of file `sstream`.

```
5.625.3.23 template<typename _CharT, typename _Traits> void std::basic_streambuf< _CharT, _Traits >::setg ( char_type
* __gbeg, char_type * __gnext, char_type * __gend ) [inline], [protected], [inherited]
```

Setting the three read area pointers.

Parameters

<code>__gbeg</code>	A pointer.
<code>__gnext</code>	A pointer.
<code>__gend</code>	A pointer.

Postcondition

`__gbeg == eback()`, `__gnext == gptr()`, and `__gend == egptr()`

Definition at line 509 of file `streambuf`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::xsgetn()`.

5.625.3.24 `template<typename _CharT, typename _Traits> void std::basic_streambuf<_CharT, _Traits>::setp (char_type * __pbeg, char_type * __pend)` `[inline]`, `[protected]`, `[inherited]`

Setting the three write area pointers.

Parameters

<code>__pbeg</code>	A pointer.
<code>__pend</code>	A pointer.

Postcondition

`__pbeg == pbase()`, `__pbeg == pptr()`, and `__pend == epptr()`

Definition at line 555 of file `streambuf`.

5.625.3.25 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::sgetc ()` `[inline]`, `[inherited]`

Getting the next character.

Returns

The next character, or `eof`.

If the input read position is available, returns that character, otherwise calls and returns `underflow()`. Does not move the read position after fetching the character.

Definition at line 338 of file `streambuf`.

Referenced by `std::istreambuf_iterator<_CharT, _Traits>::equal()`, `std::ostreambuf_iterator<_CharT, _Traits>::failed()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, and `std::basic_streambuf<_CharT, _Traits>::xsputn()`.

5.625.3.26 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf<_CharT, _Traits>::xsgetn (char_type * __s, streamsize __n)` `[inline]`, `[inherited]`

Entry point for `xsgetn`.

Parameters

$_s$	A buffer area.
$_n$	A count.

Returns `xsgetn(__s,__n)`. The effect is to fill `__s[0]` through `__s[__n-1]` with characters from the input sequence, if possible.

Definition at line 357 of file `streambuf`.

5.625.3.27 `template<typename _CharT, typename _Traits, typename _Alloc> virtual streamsize std::basic_stringbuf< _CharT, _Traits, _Alloc >::showmanyc () [inline],[protected],[virtual]`

Investigating the data available.

Returns

An estimate of the number of characters available in the input sequence, or -1.

If it returns a positive value, then successive calls to `underflow()` will not return `traits::eof()` until at least that number of characters have been supplied. If `showmanyc()` returns -1, then calls to `underflow()` or `uflow()` will fail. [27.5.2.4.3]/1

Note

Base class version does nothing, returns zero.

The standard adds that *the intention is not only that the calls [to `underflow` or `uflow`] will not return `eof()` but that they will return immediately.*

The standard adds that *the morphemes of `showmanyc` are **es-how-many-see**, not **show-manic**.*

Reimplemented from [std::basic_streambuf< _CharT, _Traits >](#).

Definition at line 211 of file `sstream`.

5.625.3.28 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf< _CharT, _Traits >::snextc () [inline],[inherited]`

Getting the next character.

Returns

The next character, or `eof`.

Calls `sbumpc()`, and if that function returns `traits::eof()`, so does this function. Otherwise, `sgetc()`.

Definition at line 298 of file `streambuf`.

Referenced by `std::ostreambuf_iterator< _CharT, _Traits >::failed()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_streambuf< _CharT, _Traits >::xsputn()`.

5.625.3.29 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf< _CharT, _Traits >::sputbackc (char_type _c) [inline],[inherited]`

Pushing characters back into the input stream.

Parameters

<code>__c</code>	The character to push back.
------------------	-----------------------------

Returns

The previous character, if possible.

Similar to `sungetc()`, but `__c` is pushed onto the stream instead of *the previous character*. If successful, the next character fetched from the input stream will be `__c`.

Definition at line 372 of file `streambuf`.

Referenced by `std::basic_istream<_CharT, _Traits>::putback()`.

5.625.3.30 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf<_CharT, _Traits>::sputc (char_type __c) [inline], [inherited]`

Entry point for all single-character output functions.

Parameters

<code>__c</code>	A character to output.
------------------	------------------------

Returns

`__c`, if possible.

One of two public output functions.

If a write position is available for the output sequence (i.e., the buffer is not full), stores `__c` in that position, increments the position, and returns `traits::to_int_type(__c)`. If a write position is not available, returns `overflow(__c)`.

Definition at line 424 of file `streambuf`.

Referenced by `std::basic_istream<_CharT, _Traits>::get()`, `std::ostreambuf_iterator<_CharT, _Traits>::operator=()`, and `std::basic_streambuf<_CharT, _Traits>::xsputn()`.

5.625.3.31 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf<_CharT, _Traits>::sputn (const char_type *__s, streamsize __n) [inline], [inherited]`

Entry point for all single-character output functions.

Parameters

<code>__s</code>	A buffer read area.
<code>__n</code>	A count.

One of two public output functions.

Returns `xspn(__s, __n)`. The effect is to write `__s[0]` through `__s[__n-1]` to the output sequence, if possible.

Definition at line 450 of file `streambuf`.

Referenced by `std::ostreambuf_iterator< _CharT, _Traits >::failed()`.

5.625.3.32 `template<typename _CharT, typename _Traits, typename _Alloc> __string_type std::basic_stringbuf< _CharT, _Traits, _Alloc >::str() const [inline]`

Copying out the string buffer.

Returns

A copy of one of the underlying sequences.

If the buffer is only created in input mode, the underlying character sequence is equal to the input sequence; otherwise, it is equal to the output sequence. [27.7.1.2]/1

Definition at line 166 of file `sstream`.

5.625.3.33 `template<typename _CharT, typename _Traits, typename _Alloc> void std::basic_stringbuf< _CharT, _Traits, _Alloc >::str(const __string_type & __s) [inline]`

Setting a new buffer.

Parameters

<code>__s</code>	The string to use as a new sequence.
------------------	--------------------------------------

Deallocates any previous stored sequence, then copies `s` to use as a new one.

Definition at line 190 of file `sstream`.

5.625.3.34 `template<typename _CharT, typename _Traits> int_type std::basic_streambuf< _CharT, _Traits >::sungetc() [inline], [inherited]`

Moving backwards in the input stream.

Returns

The previous character, if possible.

If a putback position is available, this function decrements the input pointer and returns that character. Otherwise, calls and returns `pbackfail()`. The effect is to *unget* the last character *gotten*.

Definition at line 397 of file `streambuf`.

Referenced by `std::basic_istream<_CharT, _Traits>::unget()`.

5.625.3.35 `template<typename _CharT, typename _Traits> virtual int std::basic_streambuf<_CharT, _Traits>::sync (void)`
`[inline], [protected], [virtual], [inherited]`

Synchronizes the buffer arrays with the controlled sequences.

Returns

-1 on failure.

Each derived class provides its own appropriate behavior, including the definition of *failure*.

Note

Base class version does nothing, returns zero.

Reimplemented in `std::basic_filebuf<_CharT, _Traits>`, `std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>`, `std::basic_filebuf<char_type, traits_type>`, `std::wbuffer_convert<_Codecvt, _Elem, _Tr>`, and `__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>`.

Definition at line 627 of file `streambuf`.

5.625.3.36 `template<typename _CharT, typename _Traits> virtual int_type std::basic_streambuf<_CharT, _Traits>::uflow (`
`) [inline], [protected], [virtual], [inherited]`

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function does the same thing as `underflow()`, and in fact is required to call that function. It also returns the new character, like `underflow()` does. However, this function also moves the read position forward by one.

Reimplemented in `__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>`.

Definition at line 700 of file `streambuf`.

5.625.3.37 `template<class _CharT, class _Traits, class _Alloc > basic_stringbuf< _CharT, _Traits, _Alloc >::int_type
std::basic_stringbuf< _CharT, _Traits, _Alloc >::underflow() [protected], [virtual]`

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function is called when the input buffer is exhausted (or does not exist, as buffering need not actually be done). If a buffer exists, it is *refilled*. In either case, the next available character is returned, or `traits::eof()` to indicate a null pending sequence.

For a formal definition of the pending sequence, see a good text such as Langer & Kreft, or [27.5.2.4.3]/7-14.

A functioning input streambuf can be created by overriding only this function (no buffer area will be used). For an example, see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html>

Note

Base class version does nothing, returns `eof()`.

Reimplemented from `std::basic_streambuf< _CharT, _Traits >`.

Definition at line 131 of file `sstream.tcc`.

References `std::ios_base::in`, and `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`.

Referenced by `std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow()`.

5.625.3.38 `template<typename _CharT, typename _Traits > streamsize std::basic_streambuf< _CharT, _Traits >::xsgetn (
char_type * __s, streamsize __n) [protected], [virtual], [inherited]`

Multiple character extraction.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to assign.

Returns

The number of characters assigned.

Fills `__s[0]` through `__s[__n-1]` with characters from the input sequence, as if by `sbumpc()`. Stops when either `__n` characters have been copied, or when `traits::eof()` would be copied.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented in [std::basic_filebuf<_CharT, _Traits>](#), [std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>](#), and [std::basic_filebuf<char_type, traits_type>](#).

Definition at line 46 of file streambuf.tcc.

References [std::min\(\)](#), and [std::basic_streambuf<_CharT, _Traits>::xsputn\(\)](#).

Referenced by [std::basic_filebuf<_CharT, _Traits>::xsgetn\(\)](#).

5.625.3.39 `template<typename _CharT, typename _Traits> streamsize std::basic_streambuf<_CharT, _Traits>::xsputn (const char_type * __s, streamsize __n) [protected], [virtual], [inherited]`

Multiple character insertion.

Parameters

_↵ _s	A buffer area.
_↵ _n	Maximum number of characters to write.

Returns

The number of characters written.

Writes `__s[0]` through `__s[__n-1]` to the output sequence, as if by `sputc()`. Stops when either `n` characters have been copied, or when `sputc()` would return `traits::eof()`.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

Reimplemented in [std::basic_filebuf<_CharT, _Traits>](#), [std::basic_filebuf<_CharT, encoding_char_traits<_CharT>>](#), and [std::basic_filebuf<char_type, traits_type>](#).

Definition at line 80 of file streambuf.tcc.

References [std::min\(\)](#), [std::basic_streambuf<_CharT, _Traits>::sgetc\(\)](#), [std::basic_streambuf<_CharT, _Traits>::snextc\(\)](#), and [std::basic_streambuf<_CharT, _Traits>::sputc\(\)](#).

Referenced by [std::basic_streambuf<_CharT, _Traits>::xsgetn\(\)](#), and [std::basic_filebuf<_CharT, _Traits>::xsputn\(\)](#).

5.625.4 Member Data Documentation

5.625.4.1 `template<typename _CharT, typename _Traits> locale std::basic_streambuf<_CharT, _Traits>::_M_buf_locale [protected], [inherited]`

Current locale setting.

Definition at line 192 of file streambuf.

Referenced by [std::basic_filebuf<_CharT, _Traits>::basic_filebuf\(\)](#).

5.625.4.2 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::_M_in_beg`
`[protected], [inherited]`

Start of get area.

Definition at line 184 of file streambuf.

5.625.4.3 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::_M_in_cur`
`[protected], [inherited]`

Current read area.

Definition at line 185 of file streambuf.

5.625.4.4 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::_M_in_end`
`[protected], [inherited]`

End of get area.

Definition at line 186 of file streambuf.

5.625.4.5 `template<typename _CharT, typename _Traits, typename _Alloc> ios_base::openmode std::basic_stringbuf<`
`_CharT, _Traits, _Alloc >::_M_mode [protected]`

Place to stash in || out || in | out settings for current stringbuf.

Definition at line 85 of file sstream.

5.625.4.6 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_beg`
`[protected], [inherited]`

Start of put area.

Definition at line 187 of file streambuf.

5.625.4.7 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_cur`
`[protected], [inherited]`

Current put area.

Definition at line 188 of file streambuf.

5.625.4.8 `template<typename _CharT, typename _Traits> char_type* std::basic_streambuf< _CharT, _Traits >::_M_out_end`
`[protected], [inherited]`

End of put area.

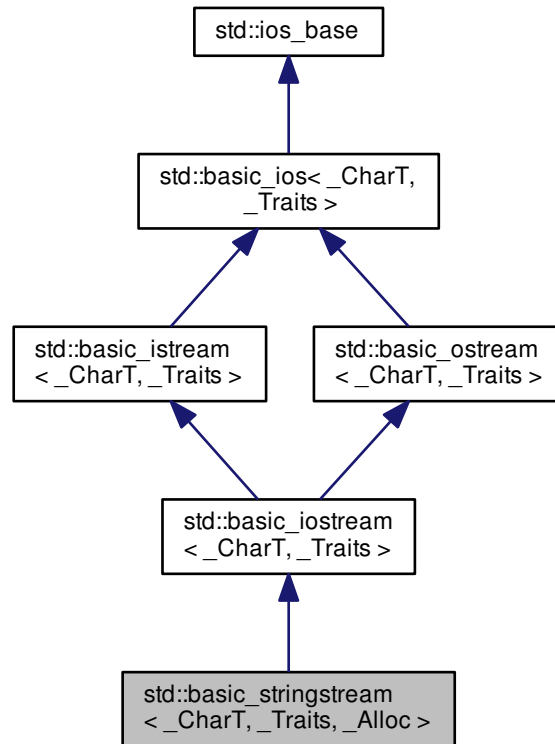
Definition at line 189 of file streambuf.

The documentation for this class was generated from the following files:

- [iosfwd](#)
- [sstream](#)
- [sstream.tcc](#)

5.626 `std::basic_stringstream<_CharT,_Traits,_Alloc>` Class Template Reference

Inheritance diagram for `std::basic_stringstream<_CharT,_Traits,_Alloc>`:



Public Types

- typedef `ctype<_CharT>` `__ctype_type`
- typedef `ctype<_CharT>` `__ctype_type`
- typedef `basic_ios<_CharT,_Traits>` `__ios_type`
- typedef `basic_ios<_CharT,_Traits>` `__ios_type`
- typedef `basic_iostream<char_type,traits_type>` `__iostream_type`
- typedef `basic_istream<_CharT,_Traits>` `__istream_type`
- typedef `num_get<_CharT,istreambuf_iterator<_CharT,_Traits>>` `__num_get_type`
- typedef `num_put<_CharT,ostreambuf_iterator<_CharT,_Traits>>` `__num_put_type`
- typedef `basic_ostream<_CharT,_Traits>` `__ostream_type`
- typedef `basic_streambuf<_CharT,_Traits>` `__streambuf_type`
- typedef `basic_streambuf<_CharT,_Traits>` `__streambuf_type`
- typedef `basic_string<_CharT,_Traits,_Alloc>` `__string_type`
- typedef `basic_stringbuf<_CharT,_Traits,_Alloc>` `__stringbuf_type`

- typedef _Alloc **allocator_type**
 - typedef _CharT **char_type**
 - enum **event** { **erase_event**, **imbue_event**, **copyfmt_event** }
 - typedef void(* **event_callback**) (event __e, ios_base & __b, int __i)
 - typedef _ios_Fmtflags **fmtflags**
 - typedef traits_type::int_type **int_type**
 - typedef int **io_state**
 - typedef _ios_istate **istate**
 - typedef traits_type::off_type **off_type**
 - typedef int **open_mode**
 - typedef _ios_Openmode **openmode**
 - typedef traits_type::pos_type **pos_type**
 - typedef int **seek_dir**
 - typedef _ios_Seekdir **seekdir**
 - typedef **std::streamoff** **streamoff**
 - typedef **std::streampos** **streampos**
 - typedef _Traits **traits_type**
-
- typedef **num_put**< _CharT, **ostreambuf_iterator**< _CharT, _Traits > > **__num_put_type**

Public Member Functions

- **basic_stringstream** (ios_base::openmode __m=ios_base::out|ios_base::in)
- **basic_stringstream** (const __string_type & __str, ios_base::openmode __m=ios_base::out|ios_base::in)
- **basic_stringstream** (const **basic_stringstream** &)=delete
- **basic_stringstream** (**basic_stringstream** && __rhs)
- **~basic_stringstream** ()
- template<typename _ValueT >
basic_istream< _CharT, _Traits > & **_M_extract** (_ValueT & __v)
- const **locale** & **_M_getloc** () const
- template<typename _ValueT >
basic_ostream< _CharT, _Traits > & **_M_insert** (_ValueT __v)
- void **_M_setstate** (iostate __state)
- bool **bad** () const
- void **clear** (iostate __state=goodbit)
- **basic_ios** & **copyfmt** (const **basic_ios** & __rhs)
- bool **eof** () const
- **iostate** **exceptions** () const
- void **exceptions** (iostate __except)
- bool **fail** () const
- char_type **fill** () const
- char_type **fill** (char_type __ch)
- **fmtflags** **flags** () const
- **fmtflags** **flags** (**fmtflags** __fmtfl)
- **__ostream_type** & **flush** ()
- **streamsize** **gcount** () const
- template<>
basic_istream< char > & **getline** (char_type * __s, **streamsize** __n, char_type __delim)

- `template<>`
`basic_istream< wchar_t > & getline (char_type *__s, streamsize __n, char_type __delim)`
 - `locale getloc () const`
 - `bool good () const`
 - `template<>`
`basic_istream< char > & ignore (streamsize __n)`
 - `template<>`
`basic_istream< char > & ignore (streamsize __n, int_type __delim)`
 - `template<>`
`basic_istream< wchar_t > & ignore (streamsize __n)`
 - `template<>`
`basic_istream< wchar_t > & ignore (streamsize __n, int_type __delim)`
 - `locale imbue (const locale &__loc)`
 - `long & iword (int __ix)`
 - `char narrow (char_type __c, char __default) const`
 - `__ostream_type & operator<< (const void *__p)`
 - `__ostream_type & operator<< (__streambuf_type *__sb)`
 - `basic_stringstream & operator= (const basic_stringstream &)=delete`
 - `basic_stringstream & operator= (basic_stringstream &&__rhs)`
 - `__istream_type & operator>> (void *__p)`
 - `__istream_type & operator>> (__streambuf_type *__sb)`
 - `streamsize precision () const`
 - `streamsize precision (streamsize __prec)`
 - `void *& pword (int __ix)`
 - `basic_streambuf< _CharT, _Traits > * rdbuf (basic_streambuf< _CharT, _Traits > *__sb)`
 - `__stringbuf_type * rdbuf () const`
 - `iosstate rdstate () const`
 - `void register_callback (event_callback __fn, int __index)`
 - `__ostream_type & seekp (pos_type)`
 - `__ostream_type & seekp (off_type, ios_base::seekdir)`
 - `fmtflags setf (fmtflags __fmtfl)`
 - `fmtflags setf (fmtflags __fmtfl, fmtflags __mask)`
 - `void setstate (iosstate __state)`
 - `__string_type str () const`
 - `void str (const __string_type &__s)`
 - `void swap (basic_stringstream &__rhs)`
 - `pos_type tellp ()`
 - `basic_ostream< _CharT, _Traits > * tie () const`
 - `basic_ostream< _CharT, _Traits > * tie (basic_ostream< _CharT, _Traits > *__tiestr)`
 - `void unsetf (fmtflags __mask)`
 - `char_type widen (char __c) const`
 - `streamsize width () const`
 - `streamsize width (streamsize __wide)`
-
- `__istream_type & operator>> (__istream_type &(*__pf)(__istream_type &))`
 - `__istream_type & operator>> (__ios_type &(*__pf)(__ios_type &))`
 - `__istream_type & operator>> (ios_base &(*__pf)(ios_base &))`

Extractors

All the `operator>>` functions (aka formatted input functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (`noskipws`) set to `false`. This has several effects, concluding with the setting of a status flag; see the `sentry` documentation for more.

If the `sentry` status is good, the function tries to extract whatever data is appropriate for the type of the argument.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- `__istream_type & operator>> (bool &__n)`
- `__istream_type & operator>> (short &__n)`
- `__istream_type & operator>> (unsigned short &__n)`
- `__istream_type & operator>> (int &__n)`
- `__istream_type & operator>> (unsigned int &__n)`
- `__istream_type & operator>> (long &__n)`
- `__istream_type & operator>> (unsigned long &__n)`
- `__istream_type & operator>> (long long &__n)`
- `__istream_type & operator>> (unsigned long long &__n)`
- `__istream_type & operator>> (float &__f)`
- `__istream_type & operator>> (double &__f)`
- `__istream_type & operator>> (long double &__f)`

Unformatted Input Functions

All the unformatted input functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_istream::sentry` with the second argument (`noskipws`) set to `true`. This has several effects, concluding with the setting of a status flag; see the `sentry` documentation for more.

If the `sentry` status is good, the function tries to extract whatever data is appropriate for the type of the argument.

The number of characters extracted is stored for later retrieval by `gcount()`.

If an exception is thrown during extraction, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- `int_type get ()`
- `__istream_type & get (char_type &__c)`
- `__istream_type & get (char_type *__s, streamsize __n, char_type __delim)`
- `__istream_type & get (char_type *__s, streamsize __n)`
- `__istream_type & get (__streambuf_type &__sb, char_type __delim)`
- `__istream_type & get (__streambuf_type &__sb)`
- `__istream_type & getline (char_type *__s, streamsize __n, char_type __delim)`
- `__istream_type & getline (char_type *__s, streamsize __n)`
- `__istream_type & ignore (streamsize __n, int_type __delim)`
- `__istream_type & ignore (streamsize __n)`
- `__istream_type & ignore ()`
- `int_type peek ()`
- `__istream_type & read (char_type *__s, streamsize __n)`
- `streamsize readsome (char_type *__s, streamsize __n)`
- `__istream_type & putback (char_type __c)`
- `__istream_type & unget ()`
- `int sync ()`
- `pos_type tellg ()`
- `__istream_type & seekg (pos_type)`
- `__istream_type & seekg (off_type, ios_base::seekdir)`
- `operator bool () const`

- `bool operator! () const`
- `__ostream_type & operator<< (__ostream_type &(__pf)(__ostream_type &))`
- `__ostream_type & operator<< (__ios_type &(__pf)(__ios_type &))`
- `__ostream_type & operator<< (ios_base &(__pf)(ios_base &))`

Inserters

All the `operator<<` functions (aka formatted output functions) have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This can have several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state without causing an `ios_base::failure` to be thrown. The original exception will then be rethrown.

- `__ostream_type & operator<< (long __n)`
- `__ostream_type & operator<< (unsigned long __n)`
- `__ostream_type & operator<< (bool __n)`
- `__ostream_type & operator<< (short __n)`
- `__ostream_type & operator<< (unsigned short __n)`
- `__ostream_type & operator<< (int __n)`
- `__ostream_type & operator<< (unsigned int __n)`
- `__ostream_type & operator<< (long long __n)`
- `__ostream_type & operator<< (unsigned long long __n)`
- `__ostream_type & operator<< (double __f)`
- `__ostream_type & operator<< (float __f)`
- `__ostream_type & operator<< (long double __f)`

Unformatted Output Functions

All the unformatted output functions have some common behavior. Each starts by constructing a temporary object of type `std::basic_ostream::sentry`. This has several effects, concluding with the setting of a status flag; see the sentry documentation for more.

If the sentry status is good, the function tries to generate whatever data is appropriate for the type of the argument.

If an exception is thrown during insertion, `ios_base::badbit` will be turned on in the stream's error state. If `badbit` is on in the stream's exceptions mask, the exception will be rethrown without completing its actions.

- `__ostream_type & put (char_type __c)`
- `void _M_write (const char_type *__s, streamsize __n)`
- `__ostream_type & write (const char_type *__s, streamsize __n)`

Static Public Member Functions

- `static bool sync_with_stdio (bool __sync=true)`
- `static int xalloc () throw ()`

Static Public Attributes

- static const `fmtflags` `adjustfield`
- static const `openmode` `app`
- static const `openmode` `ate`
- static const `iosstate` `badbit`
- static const `fmtflags` `basefield`
- static const `seekdir` `beg`
- static const `openmode` `binary`
- static const `fmtflags` `boolalpha`
- static const `seekdir` `cur`
- static const `fmtflags` `dec`
- static const `seekdir` `end`
- static const `iosstate` `eofbit`
- static const `iosstate` `failbit`
- static const `fmtflags` `fixed`
- static const `fmtflags` `floatfield`
- static const `iosstate` `goodbit`
- static const `fmtflags` `hex`
- static const `openmode` `in`
- static const `fmtflags` `internal`
- static const `fmtflags` `left`
- static const `fmtflags` `oct`
- static const `openmode` `out`
- static const `fmtflags` `right`
- static const `fmtflags` `scientific`
- static const `fmtflags` `showbase`
- static const `fmtflags` `showpoint`
- static const `fmtflags` `showpos`
- static const `fmtflags` `skipws`
- static const `openmode` `trunc`
- static const `fmtflags` `unitbuf`
- static const `fmtflags` `uppercase`

Protected Types

- enum { `_S_local_word_size` }

Protected Member Functions

- void `_M_cache_locale` (const `locale` &__loc)
- void `_M_call_callbacks` (`event` __ev) throw ()
- void `_M_dispose_callbacks` (void) throw ()
- template<typename _ValueT >
 `__istream_type` & `_M_extract` (_ValueT &__v)
- `_Words` & `_M_grow_words` (int __index, bool __iword)
- void `_M_init` () throw ()
- template<typename _ValueT >
 `__ostream_type` & `_M_insert` (_ValueT __v)

- void **_M_move** (ios_base &) noexcept
- void **_M_swap** (ios_base &__rhs) noexcept
- void **init** (basic_streambuf<_CharT, _Traits> *__sb)
- void **move** (basic_ios &__rhs)
- void **move** (basic_ios &&__rhs)
- void **set_rdbuf** (basic_streambuf<_CharT, _Traits> *__sb)
- void **swap** (basic_ostream &__rhs)
- void **swap** (basic_ios &__rhs) noexcept
- void **swap** (basic_istream &__rhs)
- void **swap** (basic_iostream &__rhs)

Protected Attributes

- _Callback_list * **_M_callbacks**
- const __ctype_type * **_M_ctype**
- iostate **_M_exception**
- char_type **_M_fill**
- bool **_M_fill_init**
- fmtflags **_M_flags**
- streamsize **_M_gcount**
- locale **_M_ios_locale**
- _Words **_M_local_word** [_S_local_word_size]
- const __num_get_type * **_M_num_get**
- const __num_put_type * **_M_num_put**
- streamsize **_M_precision**
- basic_streambuf<_CharT, _Traits> * **_M_streambuf**
- iostate **_M_streambuf_state**
- basic_ostream<_CharT, _Traits> * **_M_tie**
- streamsize **_M_width**
- _Words * **_M_word**
- int **_M_word_size**
- _Words **_M_word_zero**

5.626.1 Detailed Description

```
template<typename _CharT, typename _Traits, typename _Alloc>
class std::basic_stringstream<_CharT, _Traits, _Alloc>
```

Controlling input and output for std::string.

Template Parameters

<code>_CharT</code>	Type of character stream.
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_CharT></code> .

This class supports reading from and writing to objects of type `std::basic_string`, using the inherited functions from `std::`

::basic_iostream. To control the associated sequence, an instance of std::basic_stringbuf is used, which this page refers to as `sb`.

Definition at line 108 of file iosfwd.

5.626.2 Member Typedef Documentation

5.626.2.1 `template<typename _CharT, typename _Traits> typedef num_put<_CharT, ostreambuf_iterator<_CharT, _Traits>
> std::basic_ios<_CharT, _Traits>::__num_put_type [inherited]`

These are non-standard types.

Definition at line 89 of file basic_ios.h.

5.626.2.2 `typedef void(* std::ios_base::event_callback)(event __e, ios_base &__b, int __i) [inherited]`

The type of an event callback function.

Parameters

<code>__e</code>	One of the members of the event enum.
<code>__b</code>	Reference to the ios_base object.
<code>__i</code>	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several ios_base and basic_ios functions, specifically imbue(), copyfmt(), and ~ios().

Definition at line 504 of file ios_base.h.

5.626.2.3 `typedef _Ios_Fmtflags std::ios_base::fmtflags [inherited]`

This is a bitmask type.

`_Ios_Fmtflags` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `fmtflags` are:

- `boolalpha`
- `dec`
- `fixed`
- `hex`
- `internal`
- `left`

- oct
- right
- scientific
- showbase
- showpoint
- showpos
- skipws
- unitbuf
- uppercase
- adjustfield
- basefield
- floatfield

Definition at line 323 of file ios_base.h.

5.626.2.4 `typedef _ios_istate std::ios_base::istate` [inherited]

This is a bitmask type.

`_Ios_Istate` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `istate` are:

- badbit
- eofbit
- failbit
- goodbit

Definition at line 398 of file ios_base.h.

5.626.2.5 `typedef _ios_Openmode std::ios_base::openmode` [inherited]

This is a bitmask type.

`_Ios_Openmode` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `openmode` are:

- app
- ate
- binary
- in
- out
- trunc

Definition at line 429 of file ios_base.h.

5.626.2.6 typedef _Ios_Seekdir std::ios_base::seekdir [inherited]

This is an enumerated type.

`_Ios_Seekdir` is implementation-defined. Defined values of type `seekdir` are:

- `beg`
- `cur`, equivalent to `SEEK_CUR` in the C standard library.
- `end`, equivalent to `SEEK_END` in the C standard library.

Definition at line 461 of file `ios_base.h`.

5.626.3 Member Enumeration Documentation

5.626.3.1 enum std::ios_base::event [inherited]

The set of events that may be passed to an event callback.

`erase_event` is used during `~ios()` and `copyfmt()`. `imbue_event` is used during `imbue()`. `copyfmt_event` is used during `copyfmt()`.

Definition at line 487 of file `ios_base.h`.

5.626.4 Constructor & Destructor Documentation

5.626.4.1 template<typename _CharT, typename _Traits, typename _Alloc> std::basic_stringstream<_CharT, _Traits, _Alloc>::basic_stringstream (ios_base::openmode __m = ios_base::out | ios_base::in) [inline], [explicit]

Default constructor starts with an empty string buffer.

Parameters

<code>__m</code>	Whether the buffer can read, or write, or both.
------------------	---

Initializes `sb` using the mode from `__m`, and passes `&sb` to the base class initializer. Does not allocate any buffer.

That's a lie. We initialize the base class with `NULL`, because the string class does its own memory management.

Definition at line 690 of file `sstream`.

```
5.626.4.2  template<typename _CharT , typename _Traits , typename _Alloc > std::basic_stringstream< _CharT,  
    _Traits, _Alloc >::basic_stringstream ( const __string_type & __str, ios_base::openmode __m =  
    ios_base::out | ios_base::in ) [inline],[explicit]
```

Starts with an existing string buffer.

Parameters

<code>__str</code>	A string to copy as a starting buffer.
<code>__m</code>	Whether the buffer can read, or write, or both.

Initializes `sb` using `__str` and `__m`, and passes `&sb` to the base class initializer.

That's a lie. We initialize the base class with `NULL`, because the string class does its own memory management.

Definition at line 706 of file `sstream`.

5.626.4.3 `template<typename _CharT, typename _Traits, typename _Alloc> std::basic_stringstream<_CharT, _Traits, _Alloc>::~basic_stringstream () [inline]`

The destructor does nothing.

The buffer is deallocated by the `stringbuf` object, not the formatting stream.

Definition at line 717 of file `sstream`.

5.626.5 Member Function Documentation

5.626.5.1 `const locale& std::ios_base::_M_getloc () const [inline], [inherited]`

Locale access.

Returns

A reference to the current locale.

Like `getloc` above, but returns a reference instead of generating a copy.

Definition at line 774 of file `ios_base.h`.

Referenced by `std::time_get<_CharT, _Inlter>::do_date_order()`, `std::time_get<_CharT, _Inlter>::do_get()`, `std::money_get<_CharT, _Inlter>::do_get()`, `std::num_get<_CharT, _Inlter>::do_get()`, `std::time_get<_CharT, _Inlter>::do_get_date()`, `std::time_get<_CharT, _Inlter>::do_get_monthname()`, `std::time_get<_CharT, _Inlter>::do_get_time()`, `std::time_get<_CharT, _Inlter>::do_get_weekday()`, `std::time_get<_CharT, _Inlter>::do_get_year()`, `std::time_put<_CharT, _Outlter>::do_put()`, `std::num_put<_CharT, _Outlter>::do_put()`, `std::time_get<_CharT, _Inlter>::get()`, and `std::time_put<_CharT, _Outlter>::put()`.

5.626.5.2 `template<typename _CharT, typename _Traits> void std::basic_ostream<_CharT, _Traits>::_M_write (const char_type* __s, streamsize __n) [inline], [inherited]`

Core write functionality, without sentry.

Parameters

<code>_↔ _s</code>	The array to insert.
<code>_↔ _n</code>	Maximum number of characters to insert.

Definition at line 311 of file ostream.

Referenced by `std::basic_ostream<_CharT, _Traits>::write()`.

5.626.5.3 `template<typename _CharT, typename _Traits> bool std::basic_ios<_CharT, _Traits>::bad () const`
`[inline], [inherited]`

Fast error checking.

Returns

True if the badbit is set.

Note that other iostate flags may also be set.

Definition at line 211 of file basic_ios.h.

5.626.5.4 `template<typename _CharT, typename _Traits> void std::basic_ios<_CharT, _Traits>::clear (iostate __state =`
`goodbit) [inherited]`

[Re]sets the error state.

Parameters

<code>__state</code>	The new state flag(s) to set.
----------------------	-------------------------------

See `std::ios_base::iostate` for the possible bit values. Most users will not need to pass an argument.

Definition at line 41 of file basic_ios.tcc.

Referenced by `std::basic_ios<char, _Traits>::exceptions()`, `std::__detail::operator>>()`, `std::basic_istream<_CharT, _Traits>::putback()`, `std::basic_ios<char, _Traits>::rdstate()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_ios<char, _Traits>::setstate()`, and `std::basic_istream<_CharT, _Traits>::unget()`.

5.626.5.5 `template<typename _CharT, typename _Traits> basic_ios<_CharT, _Traits> & std::basic_ios<_CharT, _Traits>::copyfmt (const basic_ios<_CharT, _Traits> & __rhs) [inherited]`

Copies fields of `__rhs` into this.

Parameters

<code>__rhs</code>	The source values for the copies.
--------------------	-----------------------------------

Returns

Reference to this object.

All fields of `__rhs` are copied into this object except that `rdbuf()` and `rdstate()` remain unchanged. All values in the `pword` and `iword` arrays are copied. Before copying, each callback is invoked with `erase_event`. After copying, each (new) callback is invoked with `copyfmt_event`. The final step is to copy exceptions().

Definition at line 63 of file `basic_ios.tcc`.

References `std::basic_ios< _CharT, _Traits >::exceptions()`, `std::basic_ios< _CharT, _Traits >::fill()`, `std::ios_base::flags()`, `std::ios_base::getloc()`, `std::ios_base::precision()`, `std::basic_ios< _CharT, _Traits >::tie()`, `std::tie()`, and `std::ios_base::width()`.

Referenced by `std::basic_ios< char, _Traits >::rdbuf()`.

5.626.5.6 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::eof () const`
`[inline], [inherited]`

Fast error checking.

Returns

True if the eofbit is set.

Note that other `iostate` flags may also be set.

Definition at line 190 of file `basic_ios.h`.

5.626.5.7 `template<typename _CharT, typename _Traits> iostate std::basic_ios< _CharT, _Traits >::exceptions () const`
`[inline], [inherited]`

Throwing exceptions on errors.

Returns

The current exceptions mask.

This changes nothing in the stream. See the one-argument version of `exceptions(iostate)` for the meaning of the return value.

Definition at line 222 of file `basic_ios.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, and `std::basic_ios< char, _Traits >::setstate()`.

5.626.5.8 `template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits >::exceptions (iostate _except)` `[inline], [inherited]`

Throwing exceptions on errors.

Parameters

<code>__except</code>	The new exceptions mask.
-----------------------	--------------------------

By default, error flags are set silently. You can set an exceptions mask for each stream; if a bit in the mask becomes set in the error flags, then an exception of type `std::ios_base::failure` is thrown.

If the error flag is already set when the exceptions mask is added, the exception is immediately thrown. Try running the following under GCC 3.1 or later:

```

1 #include <iostream>
2 #include <fstream>
3 #include <exception>
4
5 int main()
6 {
7     std::set_terminate (__gnu_cxx::__verbose_terminate_handler);
8
9     std::ifstream f ("/etc/motd");
10
11     std::cerr << "Setting badbit\n";
12     f.setstate (std::ios_base::badbit);
13
14     std::cerr << "Setting exception mask\n";
15     f.exceptions (std::ios_base::badbit);
16 }
```

Definition at line 257 of file `basic_ios.h`.

5.626.5.9 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::fail () const`
`[inline], [inherited]`

Fast error checking.

Returns

True if either the badbit or the failbit is set.

Checking the badbit in `fail()` is historical practice. Note that other iostate flags may also be set.

Definition at line 201 of file `basic_ios.h`.

Referenced by `std::basic_ios< char, _Traits >::operator bool()`, `std::basic_ios< char, _Traits >::operator!()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, and `std::regex_traits< _Ch_type >::value()`.

5.626.5.10 `template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::fill () const`
`[inline], [inherited]`

Retrieves the *empty* character.

Returns

The current fill character.

It defaults to a space (' ') in the current locale.

Definition at line 370 of file `basic_ios.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::basic_ios< char, _Traits >::fill()`, `std::normal_distribution< _RealType >::operator()()`, `std::gamma_distribution< _RealType >::operator()()`, `std::negative_binomial_distribution< _IntType >::operator()()`, and `std::operator>>()`.

5.626.5.11 `template<typename _CharT, typename _Traits> char_type std::basic_ios< _CharT, _Traits >::fill (char_type __ch) [inline], [inherited]`

Sets a new *empty* character.

Parameters

<code>__ch</code>	The new character.
-------------------	--------------------

Returns

The previous fill character.

The fill character is used to fill out space when P+ characters have been requested (e.g., via `setw`), Q characters are actually used, and Q<P. It defaults to a space (' ') in the current locale.

Definition at line 390 of file `basic_ios.h`.

5.626.5.12 `fmtflags std::ios_base::flags () const [inline], [inherited]`

Access to format flags.

Returns

The format control flags for both input and output.

Definition at line 619 of file `ios_base.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >::discard()`, `std::money_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::operator()()`, `std::discard_block_engine< _RandomNumberEngine, __p, __r >::operator()()`, `std::shuffle_order_engine< _RandomNumberEngine, __k >::operator()()`, `std::normal_distribution< _RealType >::operator()()`, `std::gamma_distribution< _RealType >::operator()()`, `std::binomial_distribution< _IntType >::operator()()`, `std::negative_binomial_distribution< _IntType >::operator()()`, `std::poisson_distribution< _IntType >::operator()()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, `std::__detail::operator>>()`, `std::operator>>()`, `std::linear_congruential_engine< _UIntType, __a, __c, __m >::seed()`, and `std::basic_istream< _CharT, _Traits >::sentry::sentry()`.

5.626.5.13 `fmtflags std::ios_base::flags (fmtflags __fmtfl) [inline], [inherited]`

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

Definition at line 630 of file `ios_base.h`.

```
5.626.5.14 template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits> & std::basic_ostream<
    _CharT, _Traits>::flush( ) [inherited]
```

Synchronizing the stream buffer.

Returns

*this

If `rdbuf()` is a null pointer, changes nothing.

Otherwise, calls `rdbuf() -> pubsync()`, and if that returns -1, sets `badbit`.

Definition at line 211 of file `ostream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, `std::basic_ios< _CharT, _Traits>::setstate()`, and `std::basic_ostream< _CharT, _Traits>::tellp()`.

Referenced by `std::operator<<()`, and `std::basic_ostream< _CharT, _Traits>::write()`.

```
5.626.5.15 template<typename _CharT, typename _Traits> streamsize std::basic_istream< _CharT, _Traits>::gcount( )
    const [inline],[inherited]
```

Character counting.

Returns

The number of characters extracted by the previous unformatted input function dispatched for this stream.

Definition at line 269 of file `istream`.

```
5.626.5.16 template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits>::int_type
    std::basic_istream< _CharT, _Traits>::get( void ) [inherited]
```

Simple extraction.

Returns

A character, or `eof()`.

Tries to extract a character. If none are available, sets `failbit` and returns `traits::eof()`.

Definition at line 236 of file `istream.tcc`.

References `std::basic_istream< _CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream< _CharT, _Traits>::get()`, and `std::basic_istream< _CharT, _Traits>::operator>>()`.

```
5.626.5.17 template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream<
    _CharT, _Traits>::get( char_type & __c ) [inherited]
```

Simple extraction.

Parameters

<code>_↵</code>	The character in which to store data.
<code>_c</code>	

Returns

*this

Tries to extract a character and store it in `__c`. If none are available, sets failbit and returns `traits::eof()`.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 272 of file istream.tcc.

References `std::basic_istream< _CharT, _Traits >::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_↵`
`base::failbit`, `std::basic_istream< _CharT, _Traits >::get()`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >↵`
`::rdbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

5.626.5.18 `template<typename _CharT, typename _Traits > basic_istream< _CharT, _Traits > & std::basic_istream<`
`_CharT, _Traits >::get (char_type * __s, streamsize __n, char_type __delim) [inherited]`

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
<code>__n</code>	Maximum number of characters to store in <code>__s</code> .
<code>__delim</code>	A "stop" character.

Returns

*this

Characters are extracted and stored into `__s` until one of the following happens:

- `__n-1` characters are stored
- the input sequence reaches EOF
- the next character equals `__delim`, in which case the character is not extracted

If no characters are stored, failbit is set in the stream's error state.

In any case, a null character is stored into the next location in the array.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 309 of file istream.tcc.

References `std::basic_istream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::basic_istream<_CharT, _Traits>::get()`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sgetc()`, and `std::basic_streambuf<_CharT, _Traits>::snextc()`.

5.626.5.19 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::get (char_type* __s, streamsize __n) [inline], [inherited]`

Simple multiple-character extraction.

Parameters

<code>__s</code>	Pointer to an array.
<code>__n</code>	Maximum number of characters to store in <code>s</code> .

Returns

`*this`

Returns `get(__s, __n, widen("\n"))`.

Definition at line 354 of file istream.

5.626.5.20 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::get (__streambuf_type & __sb, char_type __delim) [inherited]`

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
<code>__delim</code>	A "stop" character.

Returns

`*this`

Characters are extracted and inserted into `__sb` until one of the following happens:

- the input sequence reaches EOF

- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted)
- the next character equals `__delim` (in this case, the character is not extracted)
- an exception occurs (and in this case is caught)

If no characters are stored, failbit is set in the stream's error state.

Definition at line 356 of file istream.tcc.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_ios< _CharT, _Traits >::setstate()`, `std::basic_streambuf< _CharT, _Traits >::sgetc()`, `std::basic_streambuf< _CharT, _Traits >::snextc()`, and `std::basic_streambuf< _CharT, _Traits >::sputc()`.

5.626.5.21 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits >::get (__streambuf_type & __sb)` `[inline]`, `[inherited]`

Extraction into another streambuf.

Parameters

<code>__sb</code>	A streambuf in which to store data.
-------------------	-------------------------------------

Returns

`*this`

Returns `get(__sb,widen("\n"))`.

Definition at line 387 of file istream.

5.626.5.22 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::getline (char_type * __s, streamsize __n, char_type __delim)` `[inherited]`

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.
<code>__delim</code>	A "stop" character.

Returns

`*this`

Extracts and stores characters into `__s` until one of the following happens. Note that these criteria are required to be tested in the order listed here, to allow an input line to exactly fill the `__s` array without setting failbit.

1. the input sequence reaches end-of-file, in which case eofbit is set in the stream error state
2. the next character equals `__delim`, in which case the character is extracted (and therefore counted in `gcount()`) but not stored
3. `__n-1` characters are stored, in which case failbit is set in the stream error state

If no characters are extracted, failbit is set. (An empty line of input should therefore not cause failbit to be set.)

In any case, a null character is stored in the next location in the array.

Definition at line 400 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_streambuf<_CharT, _Traits>::sbumpc()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sgetc()`, and `std::basic_streambuf<_CharT, _Traits>::snextc()`.

Referenced by `std::basic_istream<_CharT, _Traits>::get()`.

5.626.5.23 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::getline (char_type * __s, streamsize __n) [inline], [inherited]`

String extraction.

Parameters

<code>__s</code>	A character array in which to store the data.
<code>__n</code>	Maximum number of characters to extract.

Returns

`*this`

Returns `getline(__s, __n, widen("\n"))`.

Definition at line 427 of file `istream`.

5.626.5.24 `template<> basic_istream<char> & std::basic_istream<char>::getline (char_type * __s, streamsize __n, char_type __delim) [inherited]`

Explicit specialization declarations, defined in `src/istream.cc`.

5.626.5.25 `locale std::ios_base::getloc () const [inline], [inherited]`

Locale access.

Returns

A copy of the current locale.

If `imbue(loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale()`, the global C++ locale.

Definition at line 763 of file `ios_base.h`.

Referenced by `std::basic_ios< _CharT, _Traits >::copyfmt()`, `std::money_put< _CharT, _OutIter >::do_put()`, `std::operator>>()`, and `std::ws()`.

5.626.5.26 `template<typename _CharT, typename _Traits> bool std::basic_ios< _CharT, _Traits >::good () const`
`[inline], [inherited]`

Fast error checking.

Returns

True if no error flags are set.

A wrapper around `rdstate`.

Definition at line 180 of file `basic_ios.h`.

Referenced by `std::__detail::operator>>()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_istream< _CharT, _Traits >::sentry::sentry()`.

5.626.5.27 `template<typename _CharT, typename _Traits > basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::ignore (streamsize __n, int_type __delim)` `[inherited]`

Discarding characters.

Parameters

<code>__n</code>	Number of characters to discard.
<code>__delim</code>	A "stop" character.

Returns

`*this`

Extracts characters and throws them away until one of the following happens:

- if `__n != std::numeric_limits<int>::max()`, `__n` characters are extracted
- the input sequence reaches end-of-file
- the next character equals `__delim` (in this case, the character is extracted); note that this condition will never occur if `__delim` equals `traits::eof()`.

NB: Provide three overloads, instead of the single function (with defaults) mandated by the Standard: this leads to a better performing implementation, while still conforming to the Standard.

Definition at line 555 of file istream.tcc.

References `std::basic_istream< _CharT, _Traits >::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_istreambuf< _CharT, _Traits >::sbumpc()`, `std::basic_ios< _CharT, _Traits >::setstate()`, `std::basic_istreambuf< _CharT, _Traits >::sgetc()`, and `std::basic_istreambuf< _CharT, _Traits >::snextc()`.

5.626.5.28 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::ignore (streamsize __n) [inherited]`

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns `traits::eof()`.

Definition at line 493 of file istream.tcc.

References `std::basic_istream< _CharT, _Traits >::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_ios< _CharT, _Traits >::setstate()`, `std::basic_istreambuf< _CharT, _Traits >::sgetc()`, and `std::basic_istreambuf< _CharT, _Traits >::snextc()`.

5.626.5.29 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::ignore (void) [inherited]`

Simple extraction.

Returns

A character, or eof().

Tries to extract a character. If none are available, sets failbit and returns `traits::eof()`.

Definition at line 460 of file istream.tcc.

References `std::basic_istream< _CharT, _Traits >::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_istreambuf< _CharT, _Traits >::sbumpc()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

Referenced by `std::basic_istream< _CharT, _Traits >::getline()`, and `std::basic_istream< _CharT, _Traits >::ignore()`.

5.626.5.30 `template<typename _CharT, typename _Traits> locale std::basic_ios< _CharT, _Traits>::imbue (const locale & __loc) [inherited]`

Moves to a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls `ios_base::imbue(loc)`, and if a stream buffer is associated with this stream, calls that buffer's `pubimbue(loc)`.

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>.

Definition at line 114 of file `basic_ios.tcc`.

References `std::ios_base::imbue()`.

Referenced by `std::basic_ios< char, _Traits >::fill()`.

5.626.5.31 `template<typename _CharT, typename _Traits> void std::basic_ios< _CharT, _Traits >::init(basic_streambuf< _CharT, _Traits > * __sb)` `[protected]`, `[inherited]`

All setup is performed here.

This is called from the public constructor. It is not virtual and cannot be redefined.

Definition at line 126 of file `basic_ios.tcc`.

Referenced by `std::basic_ios< char, _Traits >::basic_ios()`.

5.626.5.32 `long& std::ios_base::iword(int __ix)` `[inline]`, `[inherited]`

Access to integer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to an integer associated with the index.

The `iword` function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 809 of file `ios_base.h`.

5.626.5.33 `template<typename _CharT, typename _Traits> char std::basic_ios<_CharT, _Traits>::narrow (char_type __c, char __dfault) const [inline], [inherited]`

Squeezes characters.

Parameters

<code>__c</code>	The character to narrow.
<code>__dfault</code>	The character to narrow.

Returns

The narrowed character.

Maps a character of `char_type` to a character of `char`, if possible.

Returns the result of

```
1 std::use_facet<ctype<char_type> > (getloc()) .narrow(c, dfault)
```

Additional notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>.

Definition at line 430 of file `basic_ios.h`.

5.626.5.34 `template<typename _CharT, typename _Traits> std::basic_ios<_CharT, _Traits>::operator bool () const [inline], [explicit], [inherited]`

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`.

Definition at line 117 of file `basic_ios.h`.

5.626.5.35 `template<typename _CharT, typename _Traits> bool std::basic_ios<_CharT, _Traits>::operator! () const [inline], [inherited]`

The quick-and-easy status check.

This allows you to write constructs such as `if (!a_stream) ...` and `while (a_stream) ...`.

Definition at line 125 of file `basic_ios.h`.

```
5.626.5.36 template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits
>::operator<<( __ostream_type &(*)(__ostream_type &)__pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

Definition at line 108 of file `ostream`.

Referenced by `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, and `std::basic_ostream< ↵
_CharT, _Traits >::sentry::sentry()`.

```
5.626.5.37 template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits
>::operator<<( __ios_type &(*)(__ios_type &)__pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

Definition at line 117 of file `ostream`.

```
5.626.5.38 template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits
>::operator<<( ios_base &(*)(ios_base &)__pf ) [inline], [inherited]
```

Interface for manipulators.

Manipulators such as `std::endl` and `std::hex` use these functions in constructs like `"std::cout << std::endl"`. For more information, see the `iomanip` header.

Definition at line 127 of file `ostream`.

```
5.626.5.39 template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits
>::operator<<( long __n ) [inline], [inherited]
```

Integer arithmetic inserters.

Parameters

<code>↵</code>	A variable of builtin integral type.
<code>__n</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 166 of file `ostream`.

5.626.5.40 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(unsigned long __n) [inline],[inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 170 of file `ostream`.

5.626.5.41 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(bool __n) [inline],[inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 174 of file `ostream`.

5.626.5.42 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::operator<<(short __n) [inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 92 of file `ostream.tcc`.

References `std::ios_base::basefield`, `std::ios_base::flags()`, `std::ios_base::hex`, `std::ios_base::oct`, and `std::basic_ostream<_CharT, _Traits>::operator<<()`.

5.626.5.43 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(unsigned short __n) [inline], [inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 181 of file `ostream`.

5.626.5.44 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits> & std::basic_ostream<_CharT, _Traits>::operator<<(int __n) [inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 106 of file `ostream.tcc`.

References `std::ios_base::basefield`, `std::ios_base::flags()`, `std::ios_base::hex`, `std::ios_base::oct`, and `std::basic_ostream<_CharT, _Traits>::operator<<()`.

5.626.5.45 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(unsigned int __n) [inline],[inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 192 of file `ostream`.

5.626.5.46 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(long long __n) [inline],[inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 201 of file `ostream`.

5.626.5.47 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(unsigned long long __n) [inline],[inherited]`

Integer arithmetic inserters.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 205 of file ostream.

5.626.5.48 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(double __f) [inline], [inherited]`

Floating point arithmetic inserters.

Parameters

↵	A variable of builtin floating point type.
↵	
↵	
↵	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 220 of file ostream.

5.626.5.49 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream<_CharT, _Traits>::operator<<(float __f) [inline], [inherited]`

Floating point arithmetic inserters.

Parameters

↵	A variable of builtin floating point type.
↵	
↵	
↵	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 224 of file ostream.

5.626.5.50 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(long double __f) [inline],[inherited]`

Floating point arithmetic inserters.

Parameters

<code>↔</code>	A variable of builtin floating point type.
<code>__↔</code>	
<code>↔</code>	
<code>__↔</code>	
<code>f</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 232 of file `ostream`.

5.626.5.51 `template<typename _CharT, typename _Traits> __ostream_type& std::basic_ostream< _CharT, _Traits>::operator<<(const void * __p) [inline],[inherited]`

Pointer arithmetic inserters.

Parameters

<code>__↔</code>	A variable of pointer type.
<code>__p</code>	

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to perform numeric formatting.

Definition at line 245 of file `ostream`.

5.626.5.52 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits> & std::basic_ostream< _CharT, _Traits>::operator<<(__streambuf_type * __sb) [inherited]`

Extracting from another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is NULL, the stream will set failbit in its error state.

Characters are extracted from `__sb` and inserted into `*this` until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output sequence fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs while getting a character from `__sb`, which sets failbit in the error state

If the function inserts no characters, failbit is set.

Definition at line 120 of file ostream.tcc.

References `std::ios_base::badbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ostream<_CharT, _Traits>::put()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

5.626.5.53 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (__istream_type &(*)(__istream_type &)_pf) [inline], [inherited]`

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

Definition at line 120 of file istream.

Referenced by `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::basic_istream<_CharT, _Traits>::sentry←::sentry()`, and `std::ws()`.

5.626.5.54 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (__ios_type &(*)(__ios_type &)_pf) [inline], [inherited]`

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

Definition at line 124 of file istream.

5.626.5.55 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (ios_base &(*)(ios_base &)_pf) [inline], [inherited]`

Interface for manipulators.

Manipulators such as `std::ws` and `std::dec` use these functions in constructs like `std::cin >> std::ws`. For more information, see the `io manip` header.

Definition at line 131 of file istream.

5.626.5.56 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (bool &_n) [inline], [inherited]`

Integer arithmetic extractors.

Parameters

<code>__↔ __n</code>	A variable of builtin integral type.
--------------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 168 of file `istream`.

5.626.5.57 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::operator>> (short & __n) [inherited]`

Integer arithmetic extractors.

Parameters

<code>__↔ __n</code>	A variable of builtin integral type.
--------------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 114 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::failbit`, `std::num_get< _CharT, _InIter>::get()`, `std::ios_base::goodbit`, `std::basic_istream< _CharT, _Traits>::operator>>()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

5.626.5.58 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::operator>> (unsigned short & __n) [inline], [inherited]`

Integer arithmetic extractors.

Parameters

<code>__↔ __n</code>	A variable of builtin integral type.
--------------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 175 of file `istream`.

5.626.5.59 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::operator>> (int &__n)` `[inherited]`

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 159 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::failbit`, `std::num_get<_CharT, _InIter>::get()`, `std::ios_base::goodbit`, `std::basic_istream<_CharT, _Traits>::operator>>()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

5.626.5.60 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> (unsigned int &__n)` `[inline]`, `[inherited]`

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 182 of file `istream`.

5.626.5.61 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::operator>> (long & __n) [inline], [inherited]`

Integer arithmetic extractors.

Parameters

<code>_↔</code> <code>_n</code>	A variable of builtin integral type.
------------------------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 186 of file `istream`.

```
5.626.5.62  template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> ( unsigned long &_n ) [inline],[inherited]
```

Integer arithmetic extractors.

Parameters

<code>_↔</code> <code>_n</code>	A variable of builtin integral type.
------------------------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 190 of file `istream`.

```
5.626.5.63  template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>> ( long long &_n ) [inline],[inherited]
```

Integer arithmetic extractors.

Parameters

<code>_↔</code> <code>_n</code>	A variable of builtin integral type.
------------------------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 195 of file `istream`.

5.626.5.64 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::operator>> (unsigned long long & __n) [inline], [inherited]`

Integer arithmetic extractors.

Parameters

<code>__n</code>	A variable of builtin integral type.
------------------	--------------------------------------

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 199 of file `istream`.

5.626.5.65 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::operator>> (float & __f) [inline], [inherited]`

Floating point arithmetic extractors.

Parameters

<code>__f</code>	A variable of builtin floating point type.
------------------	--

Returns

`*this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 214 of file `istream`.

5.626.5.66 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream< _CharT, _Traits>::operator>> (double & __f) [inline], [inherited]`

Floating point arithmetic extractors.

Parameters

\leftrightarrow	A variable of builtin floating point type.
$_ \leftrightarrow$	
\leftrightarrow	
$_ \leftrightarrow$	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 218 of file istream.

5.626.5.67 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>>(long double &_f) [inline], [inherited]`

Floating point arithmetic extractors.

Parameters

\leftrightarrow	A variable of builtin floating point type.
$_ \leftrightarrow$	
\leftrightarrow	
$_ \leftrightarrow$	
<i>f</i>	

Returns

*this if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 222 of file istream.

5.626.5.68 `template<typename _CharT, typename _Traits> __istream_type& std::basic_istream<_CharT, _Traits>::operator>>(void *&_p) [inline], [inherited]`

Basic arithmetic extractors.

Parameters

$_ \leftrightarrow$	A variable of pointer type.
<i>p</i>	

Returns

*`this` if successful

These functions use the stream's current locale (specifically, the `num_get` facet) to parse the input data.

Definition at line 235 of file `istream`.

5.626.5.69 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::operator>> (__streambuf_type* __sb) [inherited]`

Extracting into another streambuf.

Parameters

<code>__sb</code>	A pointer to a streambuf
-------------------	--------------------------

This function behaves like one of the basic arithmetic extractors, in that it also constructs a sentry object and has the same error handling behavior.

If `__sb` is `NULL`, the stream will set `failbit` in its error state.

Characters are extracted from this stream and inserted into the `__sb` streambuf until one of the following occurs:

- the input stream reaches end-of-file,
- insertion into the output buffer fails (in this case, the character that would have been inserted is not extracted), or
- an exception occurs (and in this case is caught)

If the function inserts no characters, `failbit` is set.

Definition at line 204 of file `istream.tcc`.

References `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::basic_istream<_CharT, _Traits>::get()`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

5.626.5.70 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits>::int_type std::basic_istream<_CharT, _Traits>::peek (void) [inherited]`

Looking ahead in the stream.

Returns

The next character, or `eof()`.

If, after constructing the sentry object, `good()` is `false`, returns `traits::eof()`. Otherwise reads but does not extract the next input character.

Definition at line 620 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_istream<_CharT, _Traits>::read()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream<_CharT, _Traits>::ignore()`.

5.626.5.71 **streamsize** std::ios_base::precision () const [inline],[inherited]

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Definition at line 689 of file ios_base.h.

Referenced by std::basic_ios< _CharT, _Traits >::copyfmt(), std::num_get< _CharT, _InIter >::do_get(), std::normal_↵distribution< _RealType >::operator>(), std::gamma_distribution< _RealType >::operator>(), std::negative_binomial_↵distribution< _IntType >::operator>(), and std::operator>>().

5.626.5.72 **streamsize** std::ios_base::precision (streamsize __prec) [inline],[inherited]

Changing flags.

Parameters

__prec	The new precision value.
--------	--------------------------

Returns

The previous value of precision().

Definition at line 698 of file ios_base.h.

5.626.5.73 **template**<typename _CharT, typename _Traits > **basic_ostream**< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::put (char_type __c) [inherited]

Simple insertion.

Parameters

↵ __c	The character to insert.
----------	--------------------------

Returns

*this

Tries to insert __c.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 149 of file ostream.tcc.

References `std::ios_base::badbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::setstate()`, and `std::basic_ostream<_CharT, _Traits>::write()`.

Referenced by `std::basic_ostream<_CharT, _Traits>::operator<<()`.

5.626.5.74 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<_CharT, _Traits>::putback(char_type __c) [inherited]`

Unextracting a single character.

Parameters

<code>__c</code>	The character to push back into the input stream.
------------------	---

Returns

`*this`

If `rdbuf()` is not null, calls `rdbuf()->sputbackc(c)`.

If `rdbuf()` is null or if `sputbackc()` fails, sets `badbit` in the error state.

Note

This function first clears `eofbit`. Since no characters are extracted, the next call to `gcount()` will return 0, as required by DR 60.

Definition at line 711 of file istream.tcc.

References `std::basic_istream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::rdstate()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sputbackc()`, and `std::basic_istream<_CharT, _Traits>::unget()`.

Referenced by `std::basic_istream<_CharT, _Traits>::readsome()`.

5.626.5.75 `void*& std::ios_base::pword(int __ix) [inline], [inherited]`

Access to void pointer array.

Parameters

<code>_↔ _ix</code>	Index into the array.
-------------------------	-----------------------

Returns

A reference to a void* associated with the index.

The pword function provides access to an array of pointers that can be used for any purpose. The array grows as required to hold the supplied index. All pointers in the array are initialized to 0.

The implementation reserves several indices. You should use xalloc to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 830 of file ios_base.h.

5.626.5.76 `template<typename _CharT, typename _Traits> basic_streambuf<_CharT, _Traits> * std::basic_ios<_CharT, _Traits>::rdbuf(basic_streambuf<_CharT, _Traits> * __sb) [inherited]`

Changing the underlying buffer.

Parameters

<code>__sb</code>	The new stream buffer.
-------------------	------------------------

Returns

The previous stream buffer.

Associates a new buffer with the current stream, and clears the error state.

Due to historical accidents which the LWG refuses to correct, the I/O library suffers from a design error: this function is hidden in derived classes by overrides of the zero-argument `rdbuf()`, which is non-virtual for hysterical raisins. As a result, you must use explicit qualifications to access this function via any derived class. For example:

```
1 std::fstream    foo;           // or some other derived type
2 std::streambuf* p = ....;
3
4 foo.ios::rdbuf(p);           // ios == basic_ios<char>
```

Definition at line 53 of file basic_ios.tcc.

5.626.5.77 `template<typename _CharT, typename _Traits, typename _Alloc> __stringbuf_type* std::basic_stringstream<_CharT, _Traits, _Alloc>::rdbuf() const [inline]`

Accessing the underlying buffer.

Returns

The current `basic_stringbuf` buffer.

This hides both signatures of `std::basic_ios::rdbuf()`.

Definition at line 757 of file sstream.

5.626.5.78 `template<typename _CharT, typename _Traits> iosstate std::basic_ios< _CharT, _Traits >::rdstate () const`
`[inline], [inherited]`

Returns the error state of the stream buffer.

Returns

A bit pattern (well, isn't everything?)

See `std::ios_base::iosstate` for the possible bit values. Most users will call one of the interpreting wrappers, e.g., `good()`.

Definition at line 137 of file `basic_ios.h`.

Referenced by `std::basic_ios< char, _Traits >::bad()`, `std::basic_ios< char, _Traits >::eof()`, `std::basic_ios< char, _Traits >::fail()`, `std::basic_ios< char, _Traits >::good()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ios< char, _Traits >::setstate()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

5.626.5.79 `template<typename _CharT, typename _Traits > basic_istream< _CharT, _Traits > & std::basic_istream< _CharT, _Traits >::read (char_type * __s, streamsize __n) [inherited]`

Extraction without delimiters.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

`*this`

If the stream state is `good()`, extracts characters and stores them into `__s` until one of the following happens:

- `__n` characters are stored
- the input sequence reaches end-of-file, in which case the error state is set to `failbit|eofbit`.

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 650 of file `istream.tcc`.

References `std::basic_istream< _CharT, _Traits >::M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_istream< _CharT, _Traits >::readsome()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

Referenced by `std::basic_istream< _CharT, _Traits >::peek()`.

5.626.5.80 `template<typename _CharT, typename _Traits> streamsize std::basic_istream<_CharT, _Traits>::readsome (char_type* __s, streamsize __n) [inherited]`

Extraction until the buffer is exhausted, but no more.

Parameters

<code>__s</code>	A character array.
<code>__n</code>	Maximum number of characters to store.

Returns

The number of characters extracted.

Extracts characters and stores them into `__s` depending on the number of characters remaining in the streambuf's buffer, `rdbuf()->in_avail()`, called A here:

- if `A == -1`, sets eofbit and extracts no characters
- if `A == 0`, extracts no characters
- if `A > 0`, extracts `min(A, n)`

The goal is to empty the current buffer, and to not request any more from the external input sequence controlled by the streambuf.

Definition at line 679 of file istream.tcc.

References `std::basic_istream<_CharT, _Traits>::_M_gcount`, `std::ios_base::badbit`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::min()`, `std::basic_istream<_CharT, _Traits>::putback()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream<_CharT, _Traits>::read()`.

5.626.5.81 `void std::ios_base::register_callback (event_callback __fn, int __index) [inherited]`

Add the callback `__fn` with parameter `__index`.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

5.626.5.82 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::seekg (pos_type __pos) [inherited]`

Changing the current read position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

`*this`

If `fail()` is not true, calls `rdbuf()->pubseekpos(__pos)`. If that function fails, sets failbit.

Note

This function first clears eofbit. It does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

Definition at line 845 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::basic_ios< _CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::basic_ios< _CharT, _Traits>::fail()`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::ios_base::in`, `std::basic_ios< _CharT, _Traits>::rdbuf()`, `std::basic_ios< _CharT, _Traits>::rdstate()`, and `std::basic_ios< _CharT, _Traits>::setstate()`.

Referenced by `std::basic_istream< _CharT, _Traits>::tellg()`.

5.626.5.83 `template<typename _CharT, typename _Traits> basic_istream< _CharT, _Traits> & std::basic_istream< _CharT, _Traits>::seekg (off_type __off, ios_base::seekdir __dir) [inherited]`

Changing the current read position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

`*this`

If `fail()` is not true, calls `rdbuf()->pubseekoff(__off, __dir)`. If that function fails, sets failbit.

Note

This function first clears eofbit. It does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

Definition at line 884 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::basic_ios< _CharT, _Traits >::clear()`, `std::ios_base::eofbit`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::ios_base::in`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, `std::basic_ios< _CharT, _Traits >::rdstate()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

5.626.5.84 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::seekp (pos_type __pos) [inherited]`

Changing the current write position.

Parameters

<code>__pos</code>	A file position object.
--------------------	-------------------------

Returns

`*this`

If `fail()` is not true, calls `rdbuf()->pubseekpos(pos)`. If that function fails, sets failbit.

Definition at line 258 of file `ostream.tcc`.

References `std::ios_base::badbit`, `std::basic_ios< _CharT, _Traits >::fail()`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::ios_base::out`, `std::basic_ios< _CharT, _Traits >::rdbuf()`, and `std::basic_ios< _CharT, _Traits >::setstate()`.

Referenced by `std::basic_ostream< _CharT, _Traits >::tellp()`.

5.626.5.85 `template<typename _CharT, typename _Traits> basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::seekp (off_type __off, ios_base::seekdir __dir) [inherited]`

Changing the current write position.

Parameters

<code>__off</code>	A file offset object.
<code>__dir</code>	The direction in which to seek.

Returns

`*this`

If `fail()` is not true, calls `rdbuf()->pubseekoff(off, dir)`. If that function fails, sets failbit.

Definition at line 290 of file ostream.tcc.

References `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::ios_base::out`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ios<_CharT, _Traits>::setstate()`.

5.626.5.86 `fmtflags std::ios_base::setf (fmtflags __fmtfl)` `[inline]`, `[inherited]`

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
----------------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set.

Definition at line 646 of file ios_base.h.

Referenced by `std::boolalpha()`, `std::dec()`, `std::fixed()`, `std::hex()`, `std::hexfloat()`, `std::left()`, `std::oct()`, `std::__detail<::operator>>()`, `std::right()`, `std::scientific()`, `std::showbase()`, `std::showpoint()`, `std::showpos()`, `std::skipws()`, `std::unitbuf()`, and `std::uppercase()`.

5.626.5.87 `fmtflags std::ios_base::setf (fmtflags __fmtfl, fmtflags __mask)` `[inline]`, `[inherited]`

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
<code>__mask</code>	The flags mask for <code>__fmtfl</code> .

Returns

The previous format control flags.

This function clears `mask` in the format flags, then sets `__fmtfl` & `mask`. An example mask is `ios_base::adjustfield`.

Definition at line 663 of file ios_base.h.

5.626.5.88 `template<typename _CharT, typename _Traits> void std::basic_ios<_CharT, _Traits>::setstate (iostate __state)` `[inline]`, `[inherited]`

Sets additional flags in the error state.

Parameters

<code>__state</code>	The additional state flag(s) to set.
----------------------	--------------------------------------

See `std::ios_base::iostate` for the possible bit values.

Definition at line 157 of file `basic_ios.h`.

Referenced by `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::tr2::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::unset()`, and `std::ws()`.

5.626.5.89 `template<typename _CharT, typename _Traits, typename _Alloc> __string_type std::basic_stringstream< _CharT, _Traits, _Alloc>::str () const` `[inline]`

Copying out the string buffer.

Returns

`rdbuf ()->str ()`

Definition at line 765 of file `sstream`.

5.626.5.90 `template<typename _CharT, typename _Traits, typename _Alloc> void std::basic_stringstream< _CharT, _Traits, _Alloc>::str (const __string_type & __s)` `[inline]`

Setting a new buffer.

Parameters

<code>__s</code>	The string to use as a new sequence.
------------------	--------------------------------------

Calls `rdbuf ()->str (s)`.

Definition at line 775 of file `sstream`.

5.626.5.91 `template<typename _CharT, typename _Traits> int std::basic_istream< _CharT, _Traits >::sync (void)` `[inherited]`

Synchronizing the stream buffer.

Returns

0 on success, -1 on failure

If `rdbuf()` is a null pointer, returns -1.

Otherwise, calls `rdbuf() -> pubsync()`, and if that returns -1, sets `badbit` and returns -1.

Otherwise, returns 0.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`.

Definition at line 781 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::goodbit`, `std::basic_streambuf<_CharT, _Traits>::pubsync()`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::setstate()`, and `std::basic_istream<_CharT, _Traits>::tellg()`.

Referenced by `std::basic_istream<_CharT, _Traits>::unset()`.

5.626.5.92 `static bool std::ios_base::sync_with_stdio(bool __sync = true)` `[static]`, `[inherited]`

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., `stdout`) and the standard C++ objects (e.g., `cout`). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstream.html#std.io.filestreams.binary>

5.626.5.93 `template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits>::pos_type std::basic_istream<_CharT, _Traits>::tellg(void)` `[inherited]`

Getting the current read position.

Returns

A file position object.

If `fail()` is not false, returns `pos_type(-1)` to indicate failure. Otherwise returns `rdbuf() -> pubseekoff(0, cur, in)`.

Note

This function does not count the number of characters extracted, if any, and therefore does not affect the next call to `gcount()`. At variance with `putback`, `unget` and `seekg`, `eofbit` is not cleared first.

Definition at line 817 of file `istream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::cur`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::ios_base::in`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_istream<_CharT, _Traits>::seekg()`.

Referenced by `std::basic_istream<_CharT, _Traits>::sync()`.

5.626.5.94 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>::pos_type
std::basic_ostream<_CharT, _Traits>::tellp() [inherited]`

Getting the current write position.

Returns

A file position object.

If `fail()` is not false, returns `pos_type(-1)` to indicate failure. Otherwise returns `rdbuf()->pubseekoff(0, cur, out)`.

Definition at line 237 of file `ostream.tcc`.

References `std::ios_base::badbit`, `std::ios_base::cur`, `std::basic_ios<_CharT, _Traits>::fail()`, `std::ios_base::out`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, and `std::basic_ostream<_CharT, _Traits>::seekp()`.

Referenced by `std::basic_ostream<_CharT, _Traits>::flush()`.

5.626.5.95 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>* std::basic_ios<_CharT,
_Traits>::tie() const [inline], [inherited]`

Fetches the current *tied* stream.

Returns

A pointer to the tied stream, or NULL if the stream is not tied.

A stream may be *tied* (or synchronized) to a second output stream. When this stream performs any I/O, the tied stream is first flushed. For example, `std::cin` is tied to `std::cout`.

Definition at line 295 of file `basic_ios.h`.

Referenced by `std::basic_ios<char, _Traits>::basic_ios()`, `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`, and `std::basic_istream<_CharT, _Traits>::sentry::sentry()`.

5.626.5.96 `template<typename _CharT, typename _Traits> basic_ostream<_CharT, _Traits>* std::basic_ios<_CharT,
_Traits>::tie(basic_ostream<_CharT, _Traits>* __tiestr) [inline], [inherited]`

Ties this stream to an output stream.

Parameters

<code>__tiestr</code>	The output stream.
-----------------------	--------------------

Returns

The previously tied output stream, or NULL if the stream was not tied.

This sets up a new tie; see `tie()` for more.

Definition at line 307 of file `basic_ios.h`.

```
5.626.5.97  template<typename _CharT, typename _Traits> basic_istream<_CharT, _Traits> & std::basic_istream<
            _CharT, _Traits>::unget ( void ) [inherited]
```

Unextracting the previous character.

Returns

`*this`

If `rdbuf()` is not null, calls `rdbuf()->sungetc()`.

If `rdbuf()` is null or if `sungetc()` fails, sets `badbit` in the error state.

Note

This function first clears `eofbit`. Since no characters are extracted, the next call to `gcount()` will return 0, as required by DR 60.

Definition at line 746 of file `istream.tcc`.

References `std::basic_istream<_CharT, _Traits>::M_gcount`, `std::ios_base::badbit`, `std::basic_ios<_CharT, _Traits>::clear()`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::basic_ios<_CharT, _Traits>::rdbuf()`, `std::basic_ios<_CharT, _Traits>::rdstate()`, `std::basic_ios<_CharT, _Traits>::setstate()`, `std::basic_streambuf<_CharT, _Traits>::sungetc()`, and `std::basic_istream<_CharT, _Traits>::sync()`.

Referenced by `std::__detail::operator>>()`, and `std::basic_istream<_CharT, _Traits>::putback()`.

```
5.626.5.98  void std::ios_base::unsetf ( fmtflags __mask ) [inline], [inherited]
```

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

Definition at line 678 of file `ios_base.h`.

Referenced by `std::defaultfloat()`, `std::noboolalpha()`, `std::noshowbase()`, `std::noshowpoint()`, `std::noshowpos()`, `std::noskipws()`, `std::nounitbuf()`, and `std::nouppercase()`.

5.626.5.99 `template<typename _CharT, typename _Traits> char_type std::basic_ios<_CharT, _Traits>::widen (char __c)`
`const [inline],[inherited]`

Widens characters.

Parameters

<code>__c</code>	The character to widen.
------------------	-------------------------

Returns

The widened character.

Maps a character of `char` to a character of `char_type`.

Returns the result of

```
1 std::use_facet<ctype<char_type>> >(getloc()).widen(c)
```

Additional I10n notes are at <http://gcc.gnu.org/onlinedocs/libstdc++/manual/localization.html>

Definition at line 449 of file `basic_ios.h`.

Referenced by `std::basic_ios<char, _Traits>::fill()`, `std::getline()`, `std::normal_distribution<_RealType>::operator()()`, `std::gamma_distribution<_RealType>::operator()()`, `std::negative_binomial_distribution<_IntType>::operator()()`, `std::tr2::operator>>()`, and `std::operator>>()`.

5.626.5.100 `streamsize std::ios_base::width () const [inline],[inherited]`

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Definition at line 712 of file `ios_base.h`.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::money_get<_CharT, _InIter>::do_get()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_put<_CharT, _OutIter>::do_put()`, and `std::operator>>()`.

5.626.5.101 `streamsize std::ios_base::width (streamsize __wide) [inline],[inherited]`

Changing flags.

Parameters

<code>__wide</code>	The new width value.
---------------------	----------------------

Returns

The previous value of `width()`.

Definition at line 721 of file `ios_base.h`.

5.626.5.102 `template<typename _CharT, typename _Traits > basic_ostream< _CharT, _Traits > & std::basic_ostream< _CharT, _Traits >::write (const char_type * __s, streamsize __n)` [inherited]

Character string insertion.

Parameters

<code>__s</code>	The array to insert.
<code>__n</code>	Maximum number of characters to insert.

Returns

`*this`

Characters are copied from `__s` and inserted into the stream until one of the following happens:

- `__n` characters are inserted
- inserting into the output sequence fails (in this case, `badbit` will be set in the stream's error state)

Note

This function is not overloaded on signed char and unsigned char.

Definition at line 183 of file `ostream.tcc`.

References `std::basic_ostream< _CharT, _Traits >::_M_write()`, `std::ios_base::badbit`, and `std::basic_ostream< _CharT, _Traits >::flush()`.

Referenced by `std::basic_ostream< _CharT, _Traits >::put()`.

5.626.5.103 static int std::ios_base::xalloc () throw) [static],[inherited]

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the `iword` and `pword` functions. The expectation is that an application calls `xalloc` in order to obtain an index in the `iword` and `pword` arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. `xalloc` is guaranteed to return an index that is safe to use in the `iword` and `pword` arrays.

5.626.6 Member Data Documentation

5.626.6.1 template<typename _CharT, typename _Traits> streamsize std::basic_istream< _CharT, _Traits >::M_gcount [protected],[inherited]

The number of characters extracted in the previous unformatted function; see `gcount()`.

Definition at line 82 of file `istream`.

Referenced by `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, and `std::basic_istream< _CharT, _Traits >::unget()`.

5.626.6.2 const fmtflags std::ios_base::adjustfield [static],[inherited]

A mask of `left|right|internal`. Useful for the 2-arg form of `setf`.

Definition at line 378 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::internal()`, `std::left()`, and `std::right()`.

5.626.6.3 const openmode std::ios_base::app [static],[inherited]

Seek to end before each write.

Definition at line 432 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

5.626.6.4 `const openmode std::ios_base::ate` `[static],[inherited]`

Open and seek to end immediately after opening.

Definition at line 435 of file `ios_base.h`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::open()`.

5.626.6.5 `const iostate std::ios_base::badbit` `[static],[inherited]`

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Definition at line 402 of file `ios_base.h`.

Referenced by `std::basic_ios<char, _Traits>::bad()`, `std::basic_ios<char, _Traits>::fail()`, `std::basic_ostream<_CharT, _Traits>::flush()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_ostream<_CharT, _Traits>::operator<<()`, `std::operator<<()`, `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::operator>>()`, `std::basic_istream<_CharT, _Traits>::peek()`, `std::basic_ostream<_CharT, _Traits>::put()`, `std::basic_istream<_CharT, _Traits>::putback()`, `std::basic_istream<_CharT, _Traits>::read()`, `std::basic_istream<_CharT, _Traits>::readsome()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_ostream<_CharT, _Traits>::seekp()`, `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`, `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_istream<_CharT, _Traits>::sync()`, `std::basic_istream<_CharT, _Traits>::tellg()`, `std::basic_ostream<_CharT, _Traits>::tellp()`, `std::basic_istream<_CharT, _Traits>::unset()`, and `std::basic_ostream<_CharT, _Traits>::write()`.

5.626.6.6 `const fmtflags std::ios_base::basefield` `[static],[inherited]`

A mask of `dec|oct|hex`. Useful for the 2-arg form of `setf`.

Definition at line 381 of file `ios_base.h`.

Referenced by `std::dec()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_put<_CharT, _OutIter>::do_put()`, `std::hex()`, `std::oct()`, and `std::basic_ostream<_CharT, _Traits>::operator<<()`.

5.626.6.7 `const seekdir std::ios_base::beg` `[static],[inherited]`

Request a seek relative to the beginning of the stream.

Definition at line 464 of file `ios_base.h`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::seekpos()`, and `__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>::sync()`.

5.626.6.8 `const openmode std::ios_base::binary` `[static],[inherited]`

Perform input and output in binary mode (as opposed to text mode). This is probably not what you think it is; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.ostream.binary>.

Definition at line 440 of file `ios_base.h`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::showmanyc()`.

5.626.6.9 const fmtflags std::ios_base::boolalpha [static], [inherited]

Insert/extract `bool` in alphabetic rather than numeric format.

Definition at line 326 of file `ios_base.h`.

Referenced by `std::boolalpha()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_put<_CharT, _OutIter>::do_put()`, and `std::noboolalpha()`.

5.626.6.10 const seekdir std::ios_base::cur [static], [inherited]

Request a seek relative to the current position within the sequence.

Definition at line 467 of file `ios_base.h`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::imbue()`, `std::basic_filebuf<_CharT, _Traits>::overflow()`, `std::basic_filebuf<_CharT, _Traits>::pbackfail()`, `std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekoff()`, `std::basic_filebuf<_CharT, _Traits>::seekoff()`, `__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>::sync()`, `std::basic_istream<_CharT, _Traits>::tellg()`, and `std::basic_ostream<_CharT, _Traits>::tellp()`.

5.626.6.11 const fmtflags std::ios_base::dec [static], [inherited]

Converts integer input or generates integer output in decimal base.

Definition at line 329 of file `ios_base.h`.

Referenced by `std::dec()`.

5.626.6.12 const seekdir std::ios_base::end [static], [inherited]

Request a seek relative to the current end of the sequence.

Definition at line 470 of file `ios_base.h`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::open()`, and `std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekoff()`.

5.626.6.13 const iostate std::ios_base::eofbit [static], [inherited]

Indicates that an input operation reached the end of an input sequence.

Definition at line 405 of file `ios_base.h`.

Referenced by `std::time_get<_CharT, _InIter>::do_get()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::time_get<_CharT, _InIter>::do_get_date()`, `std::time_get<_CharT, _InIter>::do_get_monthname()`, `std::time_get<_CharT, _InIter>::do_get_time()`, `std::time_get<_CharT, _InIter>::do_get_weekday()`, `std::time_get<_CharT, _InIter>::do_get_year()`, `std::basic_ios<char, _Traits>::eof()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::time_get<_CharT, _InIter>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::operator>>()`, `std::basic_istream<_CharT, _Traits>::peek()`, `std::basic_istream<_CharT, _Traits>::putback()`, `std::basic_istream<_CharT, _Traits>::read()`, `std::basic_istream<_CharT, _Traits>::readsome()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_istream<_CharT, _Traits>::unget()`, and `std::ws()`.

5.626.6.14 `const ios_base::failbit` `[static], [inherited]`

Indicates that an input operation failed to read the expected characters, or that an output operation failed to generate the desired characters.

Definition at line 410 of file `ios_base.h`.

Referenced by `std::time_get< _CharT, _InIter >::do_date_order()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::time_get< _CharT, _InIter >::do_get_monthname()`, `std::time_get< _CharT, _InIter >::do_get_weekday()`, `std::time_get< _CharT, _InIter >::do_get_year()`, `std::basic_ios< char, _Traits >::fail()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::time_get< _CharT, _InIter >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, and `std::basic_istream< _CharT, _Traits >::sentry::sentry()`.

5.626.6.15 `const fmtflags std::ios_base::fixed` `[static], [inherited]`

Generate floating-point output in fixed-point notation.

Definition at line 332 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _InIter >::do_get()`, `std::fixed()`, and `std::hexfloat()`.

5.626.6.16 `const fmtflags std::ios_base::floatfield` `[static], [inherited]`

A mask of `scientific|fixed`. Useful for the 2-arg form of `setf`.

Definition at line 384 of file `ios_base.h`.

Referenced by `std::defaultfloat()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::fixed()`, `std::hexfloat()`, and `std::scientific()`.

5.626.6.17 `const ios_base::goodbit` `[static], [inherited]`

Indicates all is well.

Definition at line 413 of file `ios_base.h`.

Referenced by `std::time_get< _CharT, _InIter >::do_date_order()`, `std::time_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::time_get< _CharT, _InIter >::do_get_monthname()`, `std::time_get< _CharT, _InIter >::do_get_weekday()`, `std::time_get< _CharT, _InIter >::do_get_year()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::time_get< _CharT, _InIter >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_ios< char, _Traits >::rdstate()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sync()`, and `std::basic_istream< _CharT, _Traits >::unset()`.

5.626.6.18 const fmtflags std::ios_base::hex [static], [inherited]

Converts integer input or generates integer output in hexadecimal base.

Definition at line 335 of file ios_base.h.

Referenced by std::num_get< _CharT, _InIter >::do_get(), std::num_put< _CharT, _OutIter >::do_put(), std::hex(), and std::basic_ostream< _CharT, _Traits >::operator<<().

5.626.6.19 const openmode std::ios_base::in [static], [inherited]

Open for input. Default for ifstream and fstream.

Definition at line 443 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::pbackfail(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos(), std::basic_filebuf< _CharT, _Traits >::showmanyc(), __gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync(), std::basic_istream< _CharT, _Traits >::tellg(), std::basic_stringbuf< _CharT, _Traits, _Alloc >::underflow(), std::basic_filebuf< _CharT, _Traits >::underflow(), and std::basic_filebuf< _CharT, _Traits >::xsgetn().

5.626.6.20 const fmtflags std::ios_base::internal [static], [inherited]

Adds fill characters at a designated internal point in certain generated output, or identical to right if no such point is designated.

Definition at line 340 of file ios_base.h.

Referenced by std::money_get< _CharT, _InIter >::do_get(), std::num_put< _CharT, _OutIter >::do_put(), and std::internal().

5.626.6.21 const fmtflags std::ios_base::left [static], [inherited]

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.)

Definition at line 344 of file ios_base.h.

Referenced by std::money_get< _CharT, _InIter >::do_get(), std::num_put< _CharT, _OutIter >::do_put(), and std::left().

5.626.6.22 const fmtflags std::ios_base::oct [static], [inherited]

Converts integer input or generates integer output in octal base.

Definition at line 347 of file ios_base.h.

Referenced by std::num_get< _CharT, _InIter >::do_get(), std::oct(), and std::basic_ostream< _CharT, _Traits >::operator<<().

5.626.6.23 `const openmode std::ios_base::out` `[static], [inherited]`

Open for output. Default for `ofstream` and `fstream`.

Definition at line 446 of file `ios_base.h`.

Referenced by `std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow()`, `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::pbackfail()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos()`, `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

5.626.6.24 `const fmtflags std::ios_base::right` `[static], [inherited]`

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Definition at line 351 of file `ios_base.h`.

Referenced by `std::right()`.

5.626.6.25 `const fmtflags std::ios_base::scientific` `[static], [inherited]`

Generates floating-point output in scientific notation.

Definition at line 354 of file `ios_base.h`.

Referenced by `std::hexfloat()`, and `std::scientific()`.

5.626.6.26 `const fmtflags std::ios_base::showbase` `[static], [inherited]`

Generates a prefix indicating the numeric base of generated integer output.

Definition at line 358 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::noshowbase()`, and `std::showbase()`.

5.626.6.27 `const fmtflags std::ios_base::showpoint` `[static], [inherited]`

Generates a decimal-point character unconditionally in generated floating-point output.

Definition at line 362 of file `ios_base.h`.

Referenced by `std::noshowpoint()`, and `std::showpoint()`.

5.626.6.28 `const fmtflags std::ios_base::showpos` `[static], [inherited]`

Generates a + sign in non-negative generated numeric output.

Definition at line 365 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _InIter >::do_get()`, `std::noshowpos()`, and `std::showpos()`.

5.626.6.29 `const fmtflags std::ios_base::skipws` `[static], [inherited]`

Skips leading white space before certain input operations.

Definition at line 368 of file `ios_base.h`.

Referenced by `std::noskipws()`, `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, and `std::skipws()`.

5.626.6.30 `const openmode std::ios_base::trunc` `[static], [inherited]`

Open for input. Default for `ofstream`.

Definition at line 449 of file `ios_base.h`.

5.626.6.31 `const fmtflags std::ios_base::unitbuf` `[static], [inherited]`

Flushes output after each output operation.

Definition at line 371 of file `ios_base.h`.

Referenced by `std::nounitbuf()`, and `std::unitbuf()`.

5.626.6.32 `const fmtflags std::ios_base::uppercase` `[static], [inherited]`

Replaces certain lowercase letters with their uppercase equivalents in generated output.

Definition at line 375 of file `ios_base.h`.

Referenced by `std::num_get<_CharT, _InIter>::do_get()`, `std::num_put<_CharT, _OutIter>::do_put()`, `std::noskipws()`, and `std::uppercase()`.

The documentation for this class was generated from the following files:

- [iosfwd](#)
- [sstream](#)

5.627 `std::bernoulli_distribution` Class Reference

Classes

- struct [param_type](#)

Public Types

- typedef bool [result_type](#)

Public Member Functions

- [bernoulli_distribution](#) (double __p=0.5)
- [bernoulli_distribution](#) (const [param_type](#) &__p)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void [__generate](#) (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void [__generate](#) (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- template<typename _UniformRandomNumberGenerator >
void [__generate](#) ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [result_type](#) max () const
- [result_type](#) min () const
- template<typename _UniformRandomNumberGenerator >
[result_type](#) operator() (_UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
[result_type](#) operator() (_UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- double [p](#) () const
- [param_type](#) [param](#) () const
- void [param](#) (const [param_type](#) &__param)
- void [reset](#) ()

Friends

- bool [operator==](#) (const [bernoulli_distribution](#) &__d1, const [bernoulli_distribution](#) &__d2)

5.627.1 Detailed Description

A Bernoulli random number distribution.

Generates a sequence of true and false values with likelihood p that true will come up and $(1 - p)$ that false will appear.

Definition at line 3407 of file random.h.

5.627.2 Member Typedef Documentation

5.627.2.1 typedef bool std::bernoulli_distribution::result_type

The type of the range of the distribution.

Definition at line 3411 of file random.h.

5.627.3 Constructor & Destructor Documentation

5.627.3.1 std::bernoulli_distribution::bernoulli_distribution (double __p = 0.5) [inline],[explicit]

Constructs a Bernoulli distribution with likelihood p .

Parameters

<code>_p</code>	[IN] The likelihood of a true result being returned. Must be in the interval $[0, 1]$.
-----------------	---

Definition at line 3444 of file `random.h`.

5.627.4 Member Function Documentation

5.627.4.1 `result_type std::bernoulli_distribution::max () const` `[inline]`

Returns the least upper bound value of the distribution.

Definition at line 3494 of file `random.h`.

References `std::numeric_limits<_Tp>::max()`.

5.627.4.2 `result_type std::bernoulli_distribution::min () const` `[inline]`

Returns the greatest lower bound value of the distribution.

Definition at line 3487 of file `random.h`.

References `std::numeric_limits<_Tp>::min()`.

5.627.4.3 `template<typename _UniformRandomNumberGenerator> result_type std::bernoulli_distribution::operator() (_UniformRandomNumberGenerator & __urng)` `[inline]`

Generating functions.

Definition at line 3502 of file `random.h`.

5.627.4.4 `double std::bernoulli_distribution::p () const` `[inline]`

Returns the `p` parameter of the distribution.

Definition at line 3465 of file `random.h`.

5.627.4.5 `param_type std::bernoulli_distribution::param () const` `[inline]`

Returns the parameter set of the distribution.

Definition at line 3472 of file `random.h`.

Referenced by `std::operator>>()`.

5.627.4.6 `void std::bernoulli_distribution::param (const param_type & __param)` `[inline]`

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

Definition at line 3480 of file random.h.

5.627.4.7 `void std::bernoulli_distribution::reset () [inline]`

Resets the distribution state.

Does nothing for a Bernoulli distribution.

Definition at line 3459 of file random.h.

5.627.5 Friends And Related Function Documentation

5.627.5.1 `bool operator== (const bernoulli_distribution & __d1, const bernoulli_distribution & __d2) [friend]`

Return true if two Bernoulli distributions have the same parameters.

Definition at line 3544 of file random.h.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.628 `std::bernoulli_distribution::param_type` Struct Reference

Public Types

- typedef [bernoulli_distribution](#) **distribution_type**

Public Member Functions

- **param_type** (double __p=0.5)
- double **p** () const

Friends

- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)

5.628.1 Detailed Description

Parameter type.

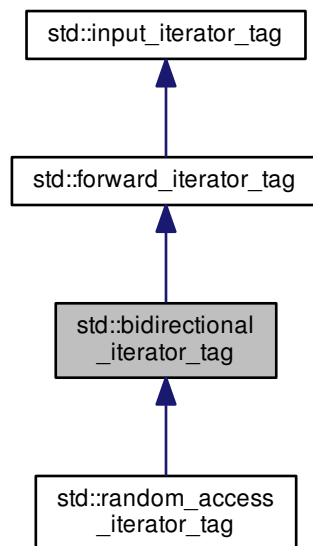
Definition at line 3413 of file `random.h`.

The documentation for this struct was generated from the following file:

- [random.h](#)

5.629 `std::bidirectional_iterator_tag` Struct Reference

Inheritance diagram for `std::bidirectional_iterator_tag`:



5.629.1 Detailed Description

Bidirectional iterators support a superset of forward iterator operations.

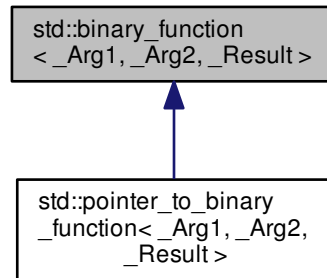
Definition at line 99 of file `stl_iterator_base_types.h`.

The documentation for this struct was generated from the following file:

- [stl_iterator_base_types.h](#)

5.630 `std::binary_function<_Arg1, _Arg2, _Result>` Struct Template Reference

Inheritance diagram for `std::binary_function<_Arg1, _Arg2, _Result>`:



Public Types

- typedef `_Arg1` [first_argument_type](#)
- typedef `_Result` [result_type](#)
- typedef `_Arg2` [second_argument_type](#)

5.630.1 Detailed Description

```
template<typename _Arg1, typename _Arg2, typename _Result>
struct std::binary_function<_Arg1, _Arg2, _Result>
```

This is one of the [functor base classes](#).

Definition at line 118 of file `stl_function.h`.

5.630.2 Member Typedef Documentation

5.630.2.1 `template<typename _Arg1, typename _Arg2, typename _Result> typedef _Arg1 std::binary_function<_Arg1, _Arg2, _Result>::first_argument_type`

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.630.2.2 `template<typename _Arg1, typename _Arg2, typename _Result> typedef _Result std::binary_function< _Arg1, _Arg2, _Result >::result_type`

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.630.2.3 `template<typename _Arg1, typename _Arg2, typename _Result> typedef _Arg2 std::binary_function< _Arg1, _Arg2, _Result >::second_argument_type`

`second_argument_type` is the type of the second argument

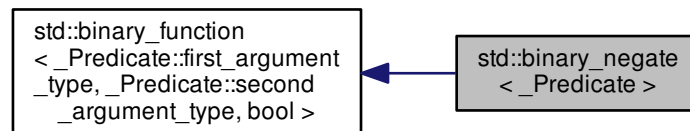
Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.631 `std::binary_negate<_Predicate>` Class Template Reference

Inheritance diagram for `std::binary_negate<_Predicate>`:



Public Types

- `typedef _Predicate::first_argument_type` [first_argument_type](#)
- `typedef bool` [result_type](#)
- `typedef _Predicate::second_argument_type` [second_argument_type](#)

Public Member Functions

- `_GLIBCXX14_CONSTEXPR` **`binary_negate`** (`const _Predicate &__x`)
- `_GLIBCXX14_CONSTEXPR` `bool` **`operator()`** (`const typename _Predicate::first_argument_type &__x, const typename _Predicate::second_argument_type &__y`) `const`

Protected Attributes

- `_Predicate_M_pred`

5.631.1 Detailed Description

```
template<typename _Predicate>
class std::binary_negate<_Predicate >
```

One of the [negation functors](#).

Definition at line 767 of file `stl_function.h`.

5.631.2 Member Typedef Documentation

5.631.2.1 `typedef _Predicate::first_argument_type std::binary_function<_Predicate::first_argument_type ,
_Predicate::second_argument_type , bool >::first_argument_type` `[inherited]`

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.631.2.2 `typedef bool std::binary_function<_Predicate::first_argument_type ,_Predicate::second_argument_type , bool
>::result_type` `[inherited]`

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.631.2.3 `typedef _Predicate::second_argument_type std::binary_function<_Predicate::first_argument_type ,
_Predicate::second_argument_type , bool >::second_argument_type` `[inherited]`

`second_argument_type` is the type of the second argument

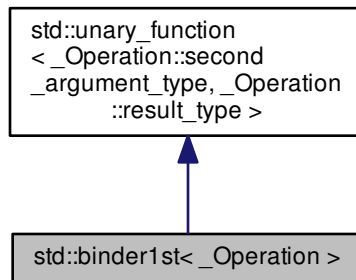
Definition at line 124 of file `stl_function.h`.

The documentation for this class was generated from the following file:

- [stl_function.h](#)

5.632 std::binder1st< _Operation > Class Template Reference

Inheritance diagram for std::binder1st< _Operation >:



Public Types

- typedef `_Operation::second_argument_type` [argument_type](#)
- typedef `_Operation::result_type` [result_type](#)

Public Member Functions

- **binder1st** (const `_Operation` &__x, const typename `_Operation::first_argument_type` &__y)
- `_Operation::result_type` **operator()** (const typename `_Operation::second_argument_type` &__x) const
- `_Operation::result_type` **operator()** (typename `_Operation::second_argument_type` &__x) const

Protected Attributes

- `_Operation` **op**
- `_Operation::first_argument_type` **value**

5.632.1 Detailed Description

```
template<typename _Operation>
class std::binder1st< _Operation >
```

One of the [binder functors](#).

Definition at line 108 of file `binders.h`.

5.632.2 Member Typedef Documentation

5.632.2.1 `typedef _Operation::second_argument_type std::unary_function< _Operation::second_argument_type ,
_Operation::result_type >::argument_type` [inherited]

`argument_type` is the type of the argument

Definition at line 108 of file `stl_function.h`.

5.632.2.2 `typedef _Operation::result_type std::unary_function< _Operation::second_argument_type , _Operation::result_type
>::result_type` [inherited]

`result_type` is the return type

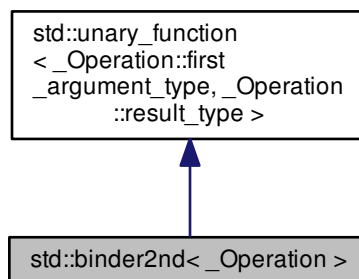
Definition at line 111 of file `stl_function.h`.

The documentation for this class was generated from the following file:

- [binders.h](#)

5.633 `std::binder2nd< _Operation >` Class Template Reference

Inheritance diagram for `std::binder2nd< _Operation >`:



Public Types

- `typedef _Operation::first_argument_type` [argument_type](#)
- `typedef _Operation::result_type` [result_type](#)

Public Member Functions

- **binder2nd** (const `_Operation` &__x, const typename `_Operation::second_argument_type` &__y)
- `_Operation::result_type` **operator()** (const typename `_Operation::first_argument_type` &__x) const
- `_Operation::result_type` **operator()** (typename `_Operation::first_argument_type` &__x) const

Protected Attributes

- `_Operation` **op**
- `_Operation::second_argument_type` **value**

5.633.1 Detailed Description

```
template<typename _Operation>
class std::binder2nd<_Operation>
```

One of the [binder functors](#).

Definition at line 143 of file `binders.h`.

5.633.2 Member Typedef Documentation

5.633.2.1 `typedef _Operation::first_argument_type std::unary_function<_Operation::first_argument_type ,
_Operation::result_type>::argument_type` `[inherited]`

`argument_type` is the type of the argument

Definition at line 108 of file `stl_function.h`.

5.633.2.2 `typedef _Operation::result_type std::unary_function<_Operation::first_argument_type ,_Operation::result_type
>::result_type` `[inherited]`

`result_type` is the return type

Definition at line 111 of file `stl_function.h`.

The documentation for this class was generated from the following file:

- [binders.h](#)

5.634 `std::binomial_distribution<_IntType>` Class Template Reference

Classes

- struct [param_type](#)

Public Types

- typedef `_IntType` `result_type`

Public Member Functions

- **binomial_distribution** (`_IntType __t=_IntType(1)`, `double __p=0.5`)
- **binomial_distribution** (const `param_type` &__p)
- template<typename `_ForwardIterator` , typename `_UniformRandomNumberGenerator` >
void **generate** (`_ForwardIterator __f`, `_ForwardIterator __t`, `_UniformRandomNumberGenerator &__urng`)
- template<typename `_ForwardIterator` , typename `_UniformRandomNumberGenerator` >
void **generate** (`_ForwardIterator __f`, `_ForwardIterator __t`, `_UniformRandomNumberGenerator &__urng`, const `param_type` &__p)
- template<typename `_UniformRandomNumberGenerator` >
void **generate** (`result_type *__f`, `result_type *__t`, `_UniformRandomNumberGenerator &__urng`, const `param_type` &__p)
- `result_type max` () const
- `result_type min` () const
- template<typename `_UniformRandomNumberGenerator` >
`result_type operator()` (`_UniformRandomNumberGenerator &__urng`)
- template<typename `_UniformRandomNumberGenerator` >
`result_type operator()` (`_UniformRandomNumberGenerator &__urng`, const `param_type` &__p)
- double `p` () const
- `param_type param` () const
- void `param` (const `param_type` &__param)
- void `reset` ()
- `_IntType t` () const

Friends

- template<typename `_IntType1` , typename `_CharT` , typename `_Traits` >
`std::basic_ostream< _CharT, _Traits > & operator<<` (`std::basic_ostream< _CharT, _Traits > &__os`, const `std::binomial_distribution< _IntType1 > &__x`)
- bool `operator==` (const `binomial_distribution` &__d1, const `binomial_distribution` &__d2)
- template<typename `_IntType1` , typename `_CharT` , typename `_Traits` >
`std::basic_istream< _CharT, _Traits > & operator>>` (`std::basic_istream< _CharT, _Traits > &__is`, `std::binomial_distribution< _IntType1 > &__x`)

5.634.1 Detailed Description

```
template<typename _IntType = int>
class std::binomial_distribution< _IntType >
```

A discrete binomial random number distribution.

The formula for the binomial probability density function is $p(i|t, p) = \binom{t}{i} p^i (1-p)^{t-i}$ where t and p are the parameters of the distribution.

Definition at line 3612 of file random.h.

5.634.2 Member Typedef Documentation

5.634.2.1 `template<typename _IntType = int> typedef _IntType std::binomial_distribution<_IntType>::result_type`

The type of the range of the distribution.

Definition at line 3615 of file `random.h`.

5.634.3 Member Function Documentation

5.634.3.1 `template<typename _IntType = int> result_type std::binomial_distribution<_IntType>::max () const`
[inline]

Returns the least upper bound value of the distribution.

Definition at line 3722 of file `random.h`.

5.634.3.2 `template<typename _IntType = int> result_type std::binomial_distribution<_IntType>::min () const`
[inline]

Returns the greatest lower bound value of the distribution.

Definition at line 3715 of file `random.h`.

5.634.3.3 `template<typename _IntType = int> template<typename _UniformRandomNumberGenerator> result_type`
`std::binomial_distribution<_IntType>::operator() (_UniformRandomNumberGenerator &__urng)` [inline]

Generating functions.

Definition at line 3730 of file `random.h`.

Referenced by `std::poisson_distribution<_IntType>::operator()()`.

5.634.3.4 `template<typename _IntType> template<typename _UniformRandomNumberGenerator> binomial_distribution<`
`_IntType>::result_type std::binomial_distribution<_IntType>::operator() (_UniformRandomNumberGenerator`
`&__urng, const param_type &__param)`

A rejection algorithm when $t * p \geq 8$ and a simple waiting time method - the second in the referenced book - otherwise.
NB: The former is available only if `_GLIBCXX_USE_C99_MATH_TR1` is defined.

Reference: Devroye, L. Non-Uniform Random Variates Generation. Springer-Verlag, New York, 1986, Ch. X, Sect. 4 (+ Errata!).

Definition at line 1534 of file `bits/random.tcc`.

References `std::abs()`, `std::dec()`, `std::numeric_limits<_Tp>::epsilon()`, `std::ios_base::flags()`, `std::left()`, `std::log()`, `std::numeric_limits<_Tp>::max()`, `std::__detail::operator>>()`, `std::binomial_distribution<_IntType>::param()`, `std::scientific()`, and `std::skipws()`.

5.634.3.5 `template<typename _IntType = int> double std::binomial_distribution<_IntType>::p () const [inline]`

Returns the distribution `p` parameter.

Definition at line 3693 of file `random.h`.

5.634.3.6 `template<typename _IntType = int> param_type std::binomial_distribution<_IntType>::param () const [inline]`

Returns the parameter set of the distribution.

Definition at line 3700 of file `random.h`.

Referenced by `std::binomial_distribution<_IntType>::operator()()`.

5.634.3.7 `template<typename _IntType = int> void std::binomial_distribution<_IntType>::param (const param_type & __param) [inline]`

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

Definition at line 3708 of file `random.h`.

5.634.3.8 `template<typename _IntType = int> void std::binomial_distribution<_IntType>::reset () [inline]`

Resets the distribution state.

Definition at line 3679 of file `random.h`.

5.634.3.9 `template<typename _IntType = int> _IntType std::binomial_distribution<_IntType>::t () const [inline]`

Returns the distribution `t` parameter.

Definition at line 3686 of file `random.h`.

5.634.4 Friends And Related Function Documentation

5.634.4.1 `template<typename _IntType = int> template<typename _IntType1, typename _CharT, typename _Traits> std::basic_ostream<_CharT, _Traits>& operator<< (std::basic_ostream<_CharT, _Traits> & __os, const std::binomial_distribution<_IntType1> & __x) [friend]`

Inserts a `binomial_distribution` random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>binomial_distribution</code> random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

5.634.4.2 `template<typename _IntType = int> bool operator== (const binomial_distribution<_IntType> &__d1, const binomial_distribution<_IntType> &__d2) [friend]`

Return true if two binomial distributions have the same parameters and the sequences that would be generated are equal.

Definition at line 3766 of file `random.h`.

5.634.4.3 `template<typename _IntType = int> template<typename _IntType1, typename _CharT, typename _Traits> std::basic_istream<_CharT, _Traits>& operator>> (std::basic_istream<_CharT, _Traits> &__is, std::binomial_distribution<_IntType1> &__x) [friend]`

Extracts a `binomial_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>binomial_distribution</code> random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.635 `std::binomial_distribution<_IntType>::param_type` Struct Reference

Public Types

- typedef `binomial_distribution<_IntType>` `distribution_type`

Public Member Functions

- **param_type** (`_IntType __t=_IntType(1), double __p=0.5`)
- double **p** () const
- `_IntType t` () const

Friends

- class **binomial_distribution**< `_IntType` >
- bool **operator==** (const `param_type` &__p1, const `param_type` &__p2)

5.635.1 Detailed Description

```
template<typename _IntType = int>
struct std::binomial_distribution< _IntType >::param_type
```

Parameter type.

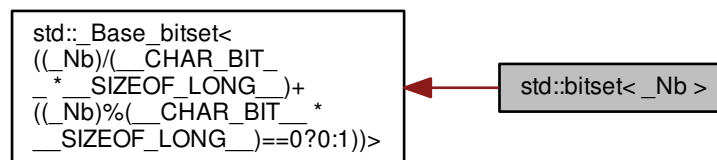
Definition at line 3621 of file random.h.

The documentation for this struct was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.636 `std::bitset< _Nb >` Class Template Reference

Inheritance diagram for `std::bitset< _Nb >`:



Classes

- class [reference](#)

Public Member Functions

- constexpr [bitset](#) () noexcept
- constexpr [bitset](#) (unsigned long long __val) noexcept
- template<class _CharT, class _Traits, class _Alloc >
[bitset](#) (const [std::basic_string](#)< _CharT, _Traits, _Alloc > &__s, size_t __position=0)
- template<class _CharT, class _Traits, class _Alloc >
[bitset](#) (const [std::basic_string](#)< _CharT, _Traits, _Alloc > &__s, size_t __position, size_t __n)
- template<class _CharT, class _Traits, class _Alloc >
[bitset](#) (const [std::basic_string](#)< _CharT, _Traits, _Alloc > &__s, size_t __position, size_t __n, _CharT __zero, _CharT __one=_CharT('1'))
- template<typename _CharT >
[bitset](#) (const _CharT * __str, typename [std::basic_string](#)< _CharT >::size_type __n=[std::basic_string](#)< _CharT >::npos, _CharT __zero=_CharT('0'), _CharT __one=_CharT('1'))
- size_t [_Find_first](#) () const noexcept
- size_t [_Find_next](#) (size_t __prev) const noexcept
- template<class _CharT, class _Traits >
void [_M_copy_from_ptr](#) (const _CharT *, size_t, size_t, size_t, _CharT, _CharT)
- template<class _CharT, class _Traits, class _Alloc >
void [_M_copy_from_string](#) (const [std::basic_string](#)< _CharT, _Traits, _Alloc > &__s, size_t __pos, size_t __n, _CharT __zero, _CharT __one)
- template<class _CharT, class _Traits, class _Alloc >
void [_M_copy_from_string](#) (const [std::basic_string](#)< _CharT, _Traits, _Alloc > &__s, size_t __pos, size_t __n)
- template<class _CharT, class _Traits, class _Alloc >
void [_M_copy_to_string](#) ([std::basic_string](#)< _CharT, _Traits, _Alloc > &, _CharT, _CharT) const
- template<class _CharT, class _Traits, class _Alloc >
void [_M_copy_to_string](#) ([std::basic_string](#)< _CharT, _Traits, _Alloc > &__s) const
- bool [all](#) () const noexcept
- bool [any](#) () const noexcept
- size_t [count](#) () const noexcept
- [bitset](#)< _Nb > & [flip](#) () noexcept
- [bitset](#)< _Nb > & [flip](#) (size_t __position)
- bool [none](#) () const noexcept
- [bitset](#)< _Nb > [operator~](#) () const noexcept
- [bitset](#)< _Nb > & [reset](#) () noexcept
- [bitset](#)< _Nb > & [reset](#) (size_t __position)
- [bitset](#)< _Nb > & [set](#) () noexcept
- [bitset](#)< _Nb > & [set](#) (size_t __position, bool __val=true)
- constexpr size_t [size](#) () const noexcept
- bool [test](#) (size_t __position) const
- template<class _CharT, class _Traits, class _Alloc >
[std::basic_string](#)< _CharT, _Traits, _Alloc > [to_string](#) () const
- template<class _CharT, class _Traits, class _Alloc >
[std::basic_string](#)< _CharT, _Traits, _Alloc > [to_string](#) (_CharT __zero, _CharT __one=_CharT('1')) const
- template<class _CharT, class _Traits >
[std::basic_string](#)< _CharT, _Traits, [std::allocator](#)< _CharT > > [to_string](#) () const
- template<class _CharT, class _Traits >
[std::basic_string](#)< _CharT, _Traits, [std::allocator](#)< _CharT > > [to_string](#) (_CharT __zero, _CharT __one=_CharT('1')) const
- template<class _CharT >
[std::basic_string](#)< _CharT, [std::char_traits](#)< _CharT >, [std::allocator](#)< _CharT > > [to_string](#) () const

- `template<class _CharT >`
`std::basic_string< _CharT, std::char_traits< _CharT >, std::allocator< _CharT > > to_string (_CharT __zero,`
`_CharT __one=_CharT('1')) const`
- `std::basic_string< char, std::char_traits< char >, std::allocator< char > > to_string () const`
- `std::basic_string< char, std::char_traits< char >, std::allocator< char > > to_string (char __zero, char __one=`
`'1') const`
- `unsigned long long to_ullong () const`
- `unsigned long to_ulong () const`
- `bitset< _Nb > & operator&= (const bitset< _Nb > &__rhs) noexcept`
- `bitset< _Nb > & operator|= (const bitset< _Nb > &__rhs) noexcept`
- `bitset< _Nb > & operator^= (const bitset< _Nb > &__rhs) noexcept`
- `bitset< _Nb > & operator<<= (size_t __position) noexcept`
- `bitset< _Nb > & operator>>= (size_t __position) noexcept`
- `bitset< _Nb > & _Unchecked_set (size_t __pos) noexcept`
- `bitset< _Nb > & _Unchecked_set (size_t __pos, int __val) noexcept`
- `bitset< _Nb > & _Unchecked_reset (size_t __pos) noexcept`
- `bitset< _Nb > & _Unchecked_flip (size_t __pos) noexcept`
- `constexpr bool _Unchecked_test (size_t __pos) const noexcept`
- `reference operator[] (size_t __position)`
- `constexpr bool operator[] (size_t __position) const`
- `bool operator== (const bitset< _Nb > &__rhs) const noexcept`
- `bool operator!= (const bitset< _Nb > &__rhs) const noexcept`
- `bitset< _Nb > operator<< (size_t __position) const noexcept`
- `bitset< _Nb > operator>> (size_t __position) const noexcept`

Private Member Functions

- `bool _M_are_all () const noexcept`
- `void _M_do_and (const _Base_bitset< _Nw > &__x) noexcept`
- `size_t _M_do_count () const noexcept`
- `size_t _M_do_find_first (size_t) const noexcept`
- `size_t _M_do_find_next (size_t, size_t) const noexcept`
- `void _M_do_flip () noexcept`
- `void _M_do_left_shift (size_t __shift) noexcept`
- `void _M_do_or (const _Base_bitset< _Nw > &__x) noexcept`
- `void _M_do_reset () noexcept`
- `void _M_do_right_shift (size_t __shift) noexcept`
- `void _M_do_set () noexcept`
- `unsigned long long _M_do_to_ullong () const`
- `unsigned long _M_do_to_ulong () const`
- `void _M_do_xor (const _Base_bitset< _Nw > &__x) noexcept`
- `const _WordT * _M_getdata () const noexcept`
- `_WordT & _M_getword (size_t __pos) noexcept`
- `constexpr _WordT _M_getword (size_t __pos) const noexcept`
- `_WordT & _M_hiword () noexcept`
- `constexpr _WordT _M_hiword () const noexcept`
- `bool _M_is_any () const noexcept`
- `bool _M_is_equal (const _Base_bitset< _Nw > &__x) const noexcept`

Static Private Member Functions

- static constexpr `_WordT _S_maskbit` (`size_t __pos`) noexcept
- static constexpr `size_t _S_whichbit` (`size_t __pos`) noexcept
- static constexpr `size_t _S_whichbyte` (`size_t __pos`) noexcept
- static constexpr `size_t _S_whichword` (`size_t __pos`) noexcept

Private Attributes

- `_WordT _M_w` [`_Nw`]

Friends

- `template<typename >`
struct **hash**
- class **reference**

5.636.1 Detailed Description

`template<size_t _Nb>`
`class std::bitset<_Nb>`

The `bitset` class represents a *fixed-size* sequence of bits.

(Note that `bitset` does *not* meet the formal requirements of a [container](#). Mainly, it lacks iterators.)

The template argument, *Nb*, may be any non-negative number, specifying the number of bits (e.g., "0", "12", "1024*1024").

In the general unoptimized case, storage is allocated in word-sized blocks. Let *B* be the number of bits in a word, then $(Nb+(B-1))/B$ words will be used for storage. $B - NbB$ bits are unused. (They are the high-order bits in the highest word.) It is a class invariant that those unused bits are always zero.

If you think of `bitset` as a *simple array of bits*, be aware that your mental picture is reversed: a `bitset` behaves the same way as bits in integers do, with the bit at index 0 in the *least significant / right-hand* position, and the bit at index *Nb-1* in the *most significant / left-hand* position. Thus, unlike other containers, a `bitset`'s index *counts from right to left*, to put it very loosely.

This behavior is preserved when translating to and from strings. For example, the first line of the following program probably prints *b('a')* is 0001100001 on a modern ASCII system.

```
1 #include <bitset>
2 #include <iostream>
3 #include <sstream>
4
5 using namespace std;
6
7 int main()
8 {
9     long    a = 'a';
10    bitset<10> b(a);
11
12    cout << "b('a') is " << b << endl;
13
14    ostringstream s;
15    s << b;
16    string str = s.str();
17    cout << "index 3 in the string is " << str[3] << " but\n"
18         << "index 3 in the bitset is " << b[3] << endl;
19 }
```

Also see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/ext_containers.html for a description of extensions.

Most of the actual code isn't contained in `bitset<>` itself, but in the base class `_Base_bitset`. The base class works with whole words, not with individual bits. This allows us to specialize `_Base_bitset` for the important special case where the `bitset` is only a single word.

Extra confusion can result due to the fact that the storage for `_Base_bitset` is a regular array, and is indexed as such. This is carefully encapsulated.

Definition at line 747 of file `bitset`.

5.636.2 Constructor & Destructor Documentation

5.636.2.1 `template<size_t_Nb> constexpr std::bitset<_Nb>::bitset() [inline],[noexcept]`

All bits set to zero.

Definition at line 861 of file `bitset`.

5.636.2.2 `template<size_t_Nb> constexpr std::bitset<_Nb>::bitset(unsigned long long __val) [inline],[noexcept]`

Initial bits bitwise-copied from a single word (others set to zero).

Definition at line 866 of file `bitset`.

5.636.2.3 `template<size_t_Nb> template<class _CharT, class _Traits, class _Alloc> std::bitset<_Nb>::bitset(const std::basic_string<_CharT, _Traits, _Alloc> &__s, size_t __position = 0) [inline],[explicit]`

Use a subset of a string.

Parameters

<code>__s</code>	A string of 0 and 1 characters.
<code>__position</code>	Index of the first character in <code>__s</code> to use; defaults to zero.

Exceptions

<code>std::out_of_range</code>	If <code>pos</code> is bigger the size of <code>__s</code> .
<code>std::invalid_argument</code>	If a character appears in the string which is neither 0 nor 1.

Definition at line 885 of file `bitset`.

5.636.2.4 `template<size_t_Nb> template<class _CharT, class _Traits, class _Alloc> std::bitset<_Nb>::bitset(const std::basic_string<_CharT, _Traits, _Alloc> &__s, size_t __position, size_t __n) [inline]`

Use a subset of a string.

Parameters

<code>__s</code>	A string of 0 and 1 characters.
<code>__position</code>	Index of the first character in <code>__s</code> to use.
<code>__n</code>	The number of characters to copy.

Exceptions

<code>std::out_of_range</code>	If <code>__position</code> is bigger the size of <code>__s</code> .
<code>std::invalid_argument</code>	If a character appears in the string which is neither 0 nor 1.

Definition at line 906 of file `bitset`.

5.636.2.5 `template<size_t _Nb> template<typename _CharT> std::bitset<_Nb>::bitset(const _CharT* __str, typename std::basic_string<_CharT>::size_type __n = std::basic_string<_CharT>::npos, _CharT __zero = _CharT('0'), _CharT __one = _CharT('1')) [inline],[explicit]`

Construct from a character array.

Parameters

<code>__str</code>	An array of characters <i>zero</i> and <i>one</i> .
<code>__n</code>	The number of characters to use.
<code>__zero</code>	The character corresponding to the value 0.
<code>__one</code>	The character corresponding to the value 1.

Exceptions

<code>std::invalid_argument</code>	If a character appears in the string which is neither <code>__zero</code> nor <code>__one</code> .
------------------------------------	--

Definition at line 938 of file `bitset`.

5.636.3 Member Function Documentation

5.636.3.1 `template<size_t _Nb> bool std::bitset<_Nb>::all() const [inline],[noexcept]`

Tests whether all the bits are on.

Returns

True if all the bits are set.

Definition at line 1326 of file `bitset`.

5.636.3.2 `template<size_t_Nb> bool std::bitset<_Nb>::any() const` `[inline],[noexcept]`

Tests whether any of the bits are on.

Returns

True if at least one bit is set.

Definition at line 1334 of file `bitset`.

5.636.3.3 `template<size_t_Nb> size_t std::bitset<_Nb>::count() const` `[inline],[noexcept]`

Returns the number of bits which are set.

Definition at line 1287 of file `bitset`.

5.636.3.4 `template<size_t_Nb> bitset<_Nb>& std::bitset<_Nb>::flip()` `[inline],[noexcept]`

Toggles every bit to its opposite value.

Definition at line 1115 of file `bitset`.

5.636.3.5 `template<size_t_Nb> bitset<_Nb>& std::bitset<_Nb>::flip(size_t__position)` `[inline]`

Toggles a given bit to its opposite value.

Parameters

<code>__position</code>	The index of the bit.
-------------------------	-----------------------

Exceptions

<code>std::out_of_range</code>	If <code>pos</code> is bigger the size of the set.
--------------------------------	--

Definition at line 1128 of file `bitset`.

5.636.3.6 `template<size_t_Nb> bool std::bitset<_Nb>::none() const` `[inline],[noexcept]`

Tests whether any of the bits are on.

Returns

True if none of the bits are set.

Definition at line 1342 of file `bitset`.

5.636.3.7 `template<size_t _Nb> bool std::bitset<_Nb>::operator!=(const bitset<_Nb> &__rhs) const` `[inline]`,
`[noexcept]`

These comparisons for equality/inequality are, well, *bitwise*.

Definition at line 1302 of file `bitset`.

5.636.3.8 `template<size_t _Nb> bitset<_Nb>& std::bitset<_Nb>::operator&= (const bitset<_Nb> &__rhs)`
`[inline]`, `[noexcept]`

Operations on bitsets.

Parameters

<code>__rhs</code>	A same-sized bitset.
--------------------	----------------------

These should be self-explanatory.

Definition at line 964 of file `bitset`.

5.636.3.9 `template<size_t _Nb> bitset<_Nb> std::bitset<_Nb>::operator<<(size_t __position) const` `[inline]`,
`[noexcept]`

Self-explanatory.

Definition at line 1348 of file `bitset`.

5.636.3.10 `template<size_t _Nb> bitset<_Nb>& std::bitset<_Nb>::operator<<= (size_t __position)` `[inline]`,
`[noexcept]`

Operations on bitsets.

Parameters

<code>__position</code>	The number of places to shift.
-------------------------	--------------------------------

These should be self-explanatory.

Definition at line 993 of file `bitset`.

5.636.3.11 `template<size_t _Nb> bool std::bitset<_Nb>::operator==(const bitset<_Nb> &__rhs) const` `[inline]`,
`[noexcept]`

These comparisons for equality/inequality are, well, *bitwise*.

Definition at line 1298 of file `bitset`.

5.636.3.12 `template<size_t _Nb> bitset<_Nb> std::bitset<_Nb>::operator>> (size_t __position) const [inline],
[noexcept]`

Self-explanatory.

Definition at line 1352 of file `bitset`.

5.636.3.13 `template<size_t _Nb> bitset<_Nb>& std::bitset<_Nb>::operator>=(size_t __position) [inline],
[noexcept]`

Operations on bitsets.

Parameters

<code>__position</code>	The number of places to shift.
-------------------------	--------------------------------

These should be self-explanatory.

Definition at line 1006 of file `bitset`.

5.636.3.14 `template<size_t _Nb> reference std::bitset<_Nb>::operator[] (size_t __position) [inline]`

Array-indexing support.

Parameters

<code>__position</code>	Index into the <code>bitset</code> .
-------------------------	--------------------------------------

Returns

A `bool` for a *const* `bitset`. For non-const `bitsets`, an instance of the reference proxy class.

Note

These operators do no range checking and throw no exceptions, as required by DR 11 to the standard.

`_GLIBCXX_RESOLVE_LIB_DEFECTS` Note that this implementation already resolves DR 11 (items 1 and 2), but does not do the range-checking required by that DR's resolution. -pme The DR has since been changed: range-checking is a precondition (users' responsibility), and these functions must not throw. -pme

Definition at line 1155 of file `bitset`.

5.636.3.15 `template<size_t _Nb> constexpr bool std::bitset<_Nb>::operator[] (size_t __position) const [inline]`

Array-indexing support.

Parameters

<code>__position</code>	Index into the bitset.
-------------------------	------------------------

Returns

A bool for a *const bitset*. For non-const bitsets, an instance of the reference proxy class.

Note

These operators do no range checking and throw no exceptions, as required by DR 11 to the standard.

_GLIBCXX_RESOLVE_LIB_DEFECTS Note that this implementation already resolves DR 11 (items 1 and 2), but does not do the range-checking required by that DR's resolution. -pme The DR has since been changed: range-checking is a precondition (users' responsibility), and these functions must not throw. -pme

Definition at line 1159 of file `bitset`.

```
5.636.3.16 template<size_t _Nb> bitset<_Nb>& std::bitset<_Nb>::operator^= ( const bitset<_Nb> & __rhs )
[inline], [noexcept]
```

Operations on bitsets.

Parameters

<code>__rhs</code>	A same-sized bitset.
--------------------	----------------------

These should be self-explanatory.

Definition at line 978 of file `bitset`.

```
5.636.3.17 template<size_t _Nb> bitset<_Nb>& std::bitset<_Nb>::operator|= ( const bitset<_Nb> & __rhs )
[inline], [noexcept]
```

Operations on bitsets.

Parameters

<code>__rhs</code>	A same-sized bitset.
--------------------	----------------------

These should be self-explanatory.

Definition at line 971 of file `bitset`.

```
5.636.3.18 template<size_t _Nb> bitset<_Nb> std::bitset<_Nb>::operator~ ( ) const [inline], [noexcept]
```

See the no-argument `flip()`.

Definition at line 1136 of file `bitset`.

5.636.3.19 `template<size_t_Nb> bitset<_Nb>& std::bitset<_Nb>::reset() [inline],[noexcept]`

Sets every bit to false.

Definition at line 1091 of file `bitset`.

5.636.3.20 `template<size_t_Nb> bitset<_Nb>& std::bitset<_Nb>::reset(size_t__position) [inline]`

Sets a given bit to false.

Parameters

<code>__position</code>	The index of the bit.
-------------------------	-----------------------

Exceptions

<code>std::out_of_range</code>	If <i>pos</i> is bigger the size of the set.
--------------------------------	--

Same as writing `set(pos, false)`.

Definition at line 1105 of file `bitset`.

5.636.3.21 `template<size_t_Nb> bitset<_Nb>& std::bitset<_Nb>::set() [inline],[noexcept]`

Sets every bit to true.

Definition at line 1067 of file `bitset`.

5.636.3.22 `template<size_t_Nb> bitset<_Nb>& std::bitset<_Nb>::set(size_t__position, bool__val=true) [inline]`

Sets a given bit to a particular value.

Parameters

<code>__position</code>	The index of the bit.
<code>__val</code>	Either true or false, defaults to true.

Exceptions

<code>std::out_of_range</code>	If <i>pos</i> is bigger the size of the set.
--------------------------------	--

Definition at line 1081 of file `bitset`.

5.636.3.23 `template<size_t_Nb> constexpr size_t std::bitset<_Nb>::size () const` `[inline],[noexcept]`

Returns the total number of bits.

Definition at line 1292 of file `bitset`.

5.636.3.24 `template<size_t_Nb> bool std::bitset<_Nb>::test (size_t__position) const` `[inline]`

Tests the value of a bit.

Parameters

<code>__position</code>	The index of a bit.
-------------------------	---------------------

Returns

The value at *pos*.

Exceptions

<code>std::out_of_range</code>	If <i>pos</i> is bigger the size of the set.
--------------------------------	--

Definition at line 1313 of file `bitset`.

5.636.3.25 `template<size_t_Nb> template<class _CharT, class _Traits, class _Alloc> std::basic_string<_CharT, _Traits, _Alloc> std::bitset<_Nb>::to_string () const` `[inline]`

Returns a character interpretation of the bitset.

Returns

The string equivalent of the bits.

Note the ordering of the bits: decreasing character positions correspond to increasing bit positions (see the main class notes for an example).

Definition at line 1189 of file `bitset`.

5.636.3.26 `template<size_t_Nb> unsigned long std::bitset<_Nb>::to_ulong () const` `[inline]`

Returns a numerical interpretation of the bitset.

Returns

The integral equivalent of the bits.

Exceptions

<code>std::overflow_error</code>	If there are too many bits to be represented in an <code>unsigned long</code> .
----------------------------------	---

Definition at line 1170 of file `bitset`.

The documentation for this class was generated from the following file:

- [bitset](#)

5.637 `std::bitset<_Nb>::reference` Class Reference

Public Member Functions

- **reference** ([bitset](#) &__b, `size_t __pos`) `noexcept`
- [reference](#) & **flip** () `noexcept`
- **operator bool** () `const noexcept`
- [reference](#) & **operator=** (`bool __x`) `noexcept`
- [reference](#) & **operator=** (`const reference &__j`) `noexcept`
- `bool operator~` () `const noexcept`

Friends

- class **bitset**

5.637.1 Detailed Description

```
template<size_t _Nb>
class std::bitset<_Nb>::reference
```

This encapsulates the concept of a single bit. An instance of this class is a proxy for an actual bit; this way the individual bit operations are done as faster word-size bitwise instructions.

Most users will never need to use this class directly; conversions to and from `bool` are automatic and should be transparent. Overloaded operators help to preserve the illusion.

(On a typical system, this *bit reference* is 64 times the size of an actual bit. Ha.)

Definition at line 798 of file `bitset`.

The documentation for this class was generated from the following file:

- [bitset](#)

5.638 std::cauchy_distribution<_RealType> Class Template Reference

Classes

- struct [param_type](#)

Public Types

- typedef _RealType [result_type](#)

Public Member Functions

- **cauchy_distribution** (_RealType __a=_RealType(0), _RealType __b=_RealType(1))
- **cauchy_distribution** (const [param_type](#) &__p)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- template<typename _UniformRandomNumberGenerator >
void **generate** ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- _RealType **a** () const
- _RealType **b** () const
- [result_type](#) **max** () const
- [result_type](#) **min** () const
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [param_type](#) **param** () const
- void **param** (const [param_type](#) &__param)
- void **reset** ()

Friends

- bool **operator==** (const [cauchy_distribution](#) &__d1, const [cauchy_distribution](#) &__d2)

5.638.1 Detailed Description

```
template<typename _RealType = double>
class std::cauchy_distribution<_RealType>
```

A `cauchy_distribution` random number distribution.

The formula for the normal probability mass function is $p(x|a, b) = (\pi b(1 + (\frac{x-a}{b})^2))^{-1}$

Definition at line 2764 of file `random.h`.

5.638.2 Member Typedef Documentation

5.638.2.1 `template<typename _RealType = double> typedef _RealType std::cauchy_distribution<_RealType>::result_type`

The type of the range of the distribution.

Definition at line 2767 of file random.h.

5.638.3 Member Function Documentation

5.638.3.1 `template<typename _RealType = double> result_type std::cauchy_distribution<_RealType>::max () const [inline]`

Returns the least upper bound value of the distribution.

Definition at line 2855 of file random.h.

References `std::numeric_limits<_Tp>::max()`.

5.638.3.2 `template<typename _RealType = double> result_type std::cauchy_distribution<_RealType>::min () const [inline]`

Returns the greatest lower bound value of the distribution.

Definition at line 2848 of file random.h.

References `std::numeric_limits<_Tp>::lowest()`.

5.638.3.3 `template<typename _RealType = double> template<typename _UniformRandomNumberGenerator> result_type std::cauchy_distribution<_RealType>::operator() (_UniformRandomNumberGenerator &__urng) [inline]`

Generating functions.

Definition at line 2863 of file random.h.

Referenced by `std::normal_distribution<_RealType>::operator()()`.

5.638.3.4 `template<typename _RealType = double> param_type std::cauchy_distribution<_RealType>::param () const [inline]`

Returns the parameter set of the distribution.

Definition at line 2833 of file random.h.

Referenced by `std::operator>>()`.

5.638.3.5 `template<typename _RealType = double> void std::cauchy_distribution<_RealType>::param (const param_type &__param) [inline]`

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

Definition at line 2841 of file random.h.

5.638.3.6 `template<typename _RealType = double> void std::cauchy_distribution<_RealType>::reset () [inline]`

Resets the distribution state.

Definition at line 2815 of file random.h.

5.638.4 Friends And Related Function Documentation

5.638.4.1 `template<typename _RealType = double> bool operator==(const cauchy_distribution<_RealType> &__d1, const cauchy_distribution<_RealType> &__d2) [friend]`

Return true if two Cauchy distributions have the same parameters.

Definition at line 2898 of file random.h.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.639 std::cauchy_distribution<_RealType>::param_type Struct Reference

Public Types

- typedef [cauchy_distribution](#)<_RealType> **distribution_type**

Public Member Functions

- **param_type** (_RealType __a=_RealType(0), _RealType __b=_RealType(1))
- _RealType **a** () const
- _RealType **b** () const

Friends

- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)

5.639.1 Detailed Description

```
template<typename _RealType = double>
struct std::cauchy_distribution< _RealType >::param_type
```

Parameter type.

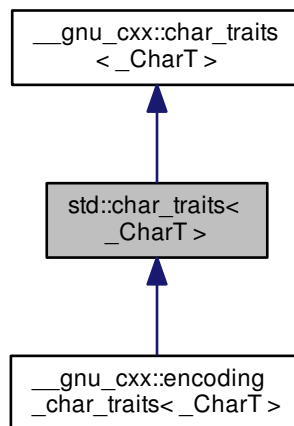
Definition at line 2773 of file random.h.

The documentation for this struct was generated from the following file:

- [random.h](#)

5.640 std::char_traits< _CharT > Struct Template Reference

Inheritance diagram for std::char_traits< _CharT >:



Public Types

- typedef `_CharT` **char_type**
- typedef `_Char_types< _CharT >::int_type` **int_type**
- typedef `_Char_types< _CharT >::off_type` **off_type**
- typedef `_Char_types< _CharT >::pos_type` **pos_type**
- typedef `_Char_types< _CharT >::state_type` **state_type**

Static Public Member Functions

- static void **assign** (char_type &__c1, const char_type &__c2)
- static char_type * **assign** (char_type *__s, std::size_t __n, char_type __a)
- static int **compare** (const char_type *__s1, const char_type *__s2, std::size_t __n)
- static char_type * **copy** (char_type *__s1, const char_type *__s2, std::size_t __n)
- static constexpr int_type **eof** ()
- static constexpr bool **eq** (const char_type &__c1, const char_type &__c2)
- static constexpr bool **eq_int_type** (const int_type &__c1, const int_type &__c2)
- static const char_type * **find** (const char_type *__s, std::size_t __n, const char_type &__a)
- static std::size_t **length** (const char_type *__s)
- static constexpr bool **lt** (const char_type &__c1, const char_type &__c2)
- static char_type * **move** (char_type *__s1, const char_type *__s2, std::size_t __n)
- static constexpr int_type **not_eof** (const int_type &__c)
- static constexpr char_type **to_char_type** (const int_type &__c)
- static constexpr int_type **to_int_type** (const char_type &__c)

5.640.1 Detailed Description

```
template<class _CharT>
struct std::char_traits<_CharT>
```

Basis for explicit traits specializations.

Note

For any given actual character type, this definition is probably wrong. Since this is just a thin wrapper around `__gnu_cxx::char_traits`, it is possible to achieve a more appropriate definition by specializing `__gnu_cxx::char_traits`.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/strings.html#strings.string.character_types for advice on how to make use of this class for *unusual* character types. Also, check out `include/ext/pod_char_traits.h`.

Definition at line 227 of file `char_traits.h`.

The documentation for this struct was generated from the following file:

- [char_traits.h](#)

5.641 `std::char_traits<__gnu_cxx::character<_Value,_Int,_St>>` Struct Template Reference

Public Types

- typedef `__gnu_cxx::character<_Value,_Int,_St>` **char_type**
- typedef char_type::int_type **int_type**
- typedef `streamoff` **off_type**
- typedef `fpos<state_type>` **pos_type**
- typedef char_type::state_type **state_type**

Static Public Member Functions

- static void **assign** ([char_type](#) &__c1, const [char_type](#) &__c2)
- static [char_type](#) * **assign** ([char_type](#) * __s, size_t __n, [char_type](#) __a)
- static int **compare** (const [char_type](#) * __s1, const [char_type](#) * __s2, size_t __n)
- static [char_type](#) * **copy** ([char_type](#) * __s1, const [char_type](#) * __s2, size_t __n)
- static int_type **eof** ()
- static bool **eq** (const [char_type](#) &__c1, const [char_type](#) &__c2)
- static bool **eq_int_type** (const int_type &__c1, const int_type &__c2)
- static const [char_type](#) * **find** (const [char_type](#) * __s, size_t __n, const [char_type](#) &__a)
- static size_t **length** (const [char_type](#) * __s)
- static bool **lt** (const [char_type](#) &__c1, const [char_type](#) &__c2)
- static [char_type](#) * **move** ([char_type](#) * __s1, const [char_type](#) * __s2, size_t __n)
- static int_type **not_eof** (const int_type &__c)
- static [char_type](#) **to_char_type** (const int_type &__i)
- static int_type **to_int_type** (const [char_type](#) &__c)

5.641.1 Detailed Description

```
template<typename _Value, typename _Int, typename _St>
struct std::char_traits< __gnu_cxx::character< _Value, _Int, _St > >
```

`char_traits<__gnu_cxx::character>` specialization.

Definition at line 97 of file `pod_char_traits.h`.

The documentation for this struct was generated from the following file:

- [pod_char_traits.h](#)

5.642 `std::char_traits< char >` Struct Template Reference

Public Types

- typedef char **char_type**
- typedef int **int_type**
- typedef [streamoff](#) **off_type**
- typedef [streampos](#) **pos_type**
- typedef mbstate_t **state_type**

Static Public Member Functions

- static void **assign** (char_type &__c1, const char_type &__c2) noexcept
- static char_type * **assign** (char_type *__s, size_t __n, char_type __a)
- static int **compare** (const char_type *__s1, const char_type *__s2, size_t __n)
- static char_type * **copy** (char_type *__s1, const char_type *__s2, size_t __n)
- static constexpr int_type **eof** () noexcept
- static constexpr bool **eq** (const char_type &__c1, const char_type &__c2) noexcept
- static constexpr bool **eq_int_type** (const int_type &__c1, const int_type &__c2) noexcept
- static const char_type * **find** (const char_type *__s, size_t __n, const char_type &__a)
- static size_t **length** (const char_type *__s)
- static constexpr bool **lt** (const char_type &__c1, const char_type &__c2) noexcept
- static char_type * **move** (char_type *__s1, const char_type *__s2, size_t __n)
- static constexpr int_type **not_eof** (const int_type &__c) noexcept
- static constexpr char_type **to_char_type** (const int_type &__c) noexcept
- static constexpr int_type **to_int_type** (const char_type &__c) noexcept

5.642.1 Detailed Description

```
template<>
struct std::char_traits< char >
```

21.1.3.1 char_traits specializations

Definition at line 233 of file char_traits.h.

The documentation for this struct was generated from the following file:

- [char_traits.h](#)

5.643 std::char_traits< wchar_t > Struct Template Reference

Public Types

- typedef wchar_t **char_type**
- typedef wint_t **int_type**
- typedef [streamoff](#) **off_type**
- typedef [wstreampos](#) **pos_type**
- typedef mbstate_t **state_type**

Static Public Member Functions

- static void **assign** (char_type &__c1, const char_type &__c2) noexcept
- static char_type * **assign** (char_type *__s, size_t __n, char_type __a)
- static int **compare** (const char_type *__s1, const char_type *__s2, size_t __n)
- static char_type * **copy** (char_type *__s1, const char_type *__s2, size_t __n)
- static constexpr int_type **eof** () noexcept
- static constexpr bool **eq** (const char_type &__c1, const char_type &__c2) noexcept
- static constexpr bool **eq_int_type** (const int_type &__c1, const int_type &__c2) noexcept
- static const char_type * **find** (const char_type *__s, size_t __n, const char_type &__a)
- static size_t **length** (const char_type *__s)
- static constexpr bool **lt** (const char_type &__c1, const char_type &__c2) noexcept
- static char_type * **move** (char_type *__s1, const char_type *__s2, size_t __n)
- static constexpr int_type **not_eof** (const int_type &__c) noexcept
- static constexpr char_type **to_char_type** (const int_type &__c) noexcept
- static constexpr int_type **to_int_type** (const char_type &__c) noexcept

5.643.1 Detailed Description

```
template<>
struct std::char_traits< wchar_t >
```

21.1.3.2 char_traits specializations

Definition at line 328 of file char_traits.h.

The documentation for this struct was generated from the following file:

- [char_traits.h](#)

5.644 std::chi_squared_distribution<_RealType> Class Template Reference

Classes

- struct [param_type](#)

Public Types

- typedef _RealType [result_type](#)

Public Member Functions

- **chi_squared_distribution** (_RealType __n=_RealType(1))
- **chi_squared_distribution** (const [param_type](#) &__p)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **__generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **__generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- template<typename _UniformRandomNumberGenerator >
void **__generate** ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
void **__generate** ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [result_type](#) **max** () const
- [result_type](#) **min** () const
- _RealType **n** () const
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [param_type](#) **param** () const
- void **param** (const [param_type](#) &__param)
- void **reset** ()

Friends

- template<typename _RealType1, typename _CharT, typename _Traits >
[std::basic_ostream](#)< _CharT, _Traits > & **operator<<** ([std::basic_ostream](#)< _CharT, _Traits > &__os, const [std::chi_squared_distribution](#)< _RealType1 > &__x)
- bool **operator==** (const [chi_squared_distribution](#) &__d1, const [chi_squared_distribution](#) &__d2)
- template<typename _RealType1, typename _CharT, typename _Traits >
[std::basic_istream](#)< _CharT, _Traits > & **operator>>** ([std::basic_istream](#)< _CharT, _Traits > &__is, [std::chi_squared_distribution](#)< _RealType1 > &__x)

5.644.1 Detailed Description

```
template<typename _RealType = double>
class std::chi_squared_distribution<_RealType>
```

A chi_squared_distribution random number distribution.

The formula for the normal probability mass function is $p(x|n) = \frac{x^{(n/2)-1} e^{-x/2}}{\Gamma(n/2) 2^{n/2}}$

Definition at line 2554 of file random.h.

5.644.2 Member Typedef Documentation

5.644.2.1 `template<typename _RealType = double> typedef _RealType std::chi_squared_distribution< _RealType >::result_type`

The type of the range of the distribution.

Definition at line 2557 of file random.h.

5.644.3 Member Function Documentation

5.644.3.1 `template<typename _RealType = double> result_type std::chi_squared_distribution< _RealType >::max ()
const [inline]`

Returns the least upper bound value of the distribution.

Definition at line 2634 of file random.h.

References `std::numeric_limits< _Tp >::max()`.

5.644.3.2 `template<typename _RealType = double> result_type std::chi_squared_distribution< _RealType >::min ()
const [inline]`

Returns the greatest lower bound value of the distribution.

Definition at line 2627 of file random.h.

5.644.3.3 `template<typename _RealType = double> template<typename _UniformRandomNumberGenerator > result_type
std::chi_squared_distribution< _RealType >::operator() (_UniformRandomNumberGenerator & __urng)
[inline]`

Generating functions.

Definition at line 2642 of file random.h.

5.644.3.4 `template<typename _RealType = double> param_type std::chi_squared_distribution< _RealType >::param ()
const [inline]`

Returns the parameter set of the distribution.

Definition at line 2612 of file random.h.

Referenced by `std::normal_distribution< _RealType >::operator()()`.

5.644.3.5 `template<typename _RealType = double> void std::chi_squared_distribution< _RealType >::param (const
param_type & __param) [inline]`

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

Definition at line 2620 of file random.h.

5.644.3.6 `template<typename _RealType = double> void std::chi_squared_distribution<_RealType>::reset ()`
`[inline]`

Resets the distribution state.

Definition at line 2598 of file random.h.

5.644.4 Friends And Related Function Documentation

5.644.4.1 `template<typename _RealType = double> template<typename _RealType1, typename _CharT, typename _Traits>`
`std::basic_ostream<_CharT, _Traits>& operator<< (std::basic_ostream<_CharT, _Traits> & __os, const`
`std::chi_squared_distribution<_RealType1> & __x) [friend]`

Inserts a chi_squared_distribution random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A chi_squared_distribution random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

5.644.4.2 `template<typename _RealType = double> bool operator==(const chi_squared_distribution<_RealType> & __d1,`
`const chi_squared_distribution<_RealType> & __d2) [friend]`

Return true if two Chi-squared distributions have the same parameters and the sequences that would be generated are equal.

Definition at line 2693 of file random.h.

5.644.4.3 `template<typename _RealType = double> template<typename _RealType1, typename _CharT, typename _Traits>`
`> std::basic_istream<_CharT, _Traits>& operator>> (std::basic_istream<_CharT, _Traits> & __is,`
`std::chi_squared_distribution<_RealType1> & __x) [friend]`

Extracts a chi_squared_distribution random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>chi_squared_distribution</code> random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.645 `std::chi_squared_distribution<_RealType>::param_type` Struct Reference**Public Types**

- typedef `chi_squared_distribution<_RealType>` **distribution_type**

Public Member Functions

- **param_type** (`_RealType __n=_RealType(1)`)
- `_RealType n` () const

Friends

- bool **operator==** (const `param_type` &__p1, const `param_type` &__p2)

5.645.1 Detailed Description

```
template<typename _RealType = double>
struct std::chi_squared_distribution<_RealType>::param_type
```

Parameter type.

Definition at line 2563 of file `random.h`.

The documentation for this struct was generated from the following file:

- [random.h](#)

5.646 std::chrono::_V2::steady_clock Struct Reference

Public Types

- typedef [chrono::nanoseconds](#) **duration**
- typedef duration::period **period**
- typedef duration::rep **rep**
- typedef [chrono::time_point](#)< [steady_clock](#), [duration](#) > **time_point**

Static Public Member Functions

- static [time_point](#) **now** () noexcept

Static Public Attributes

- static constexpr bool **is_steady**

5.646.1 Detailed Description

Monotonic clock.

Time returned has the property of only increasing at a uniform rate.

Definition at line 755 of file chrono.

The documentation for this struct was generated from the following file:

- [chrono](#)

5.647 std::chrono::_V2::system_clock Struct Reference

Public Types

- typedef [chrono::nanoseconds](#) **duration**
- typedef duration::period **period**
- typedef duration::rep **rep**
- typedef [chrono::time_point](#)< [system_clock](#), [duration](#) > **time_point**

Static Public Member Functions

- static [time_point](#) **from_time_t** (std::time_t __t) noexcept
- static [time_point](#) **now** () noexcept
- static std::time_t **to_time_t** (const [time_point](#) &__t) noexcept

Static Public Attributes

- static constexpr bool **is_steady**

5.647.1 Detailed Description

System clock.

Time returned represents wall time from the system-wide clock.

Definition at line 716 of file chrono.

The documentation for this struct was generated from the following file:

- [chrono](#)

5.648 std::chrono::duration<_Rep, _Period> Struct Template Reference

Public Types

- typedef _Period **period**
- typedef _Rep **rep**

Public Member Functions

- **duration** (const [duration](#) &)=default
- template<typename _Rep2 , typename = typename enable_if<is_convertible<_Rep2, rep>::value && (treat_as_floating_point<rep><→ ::value || !treat_as_floating_point<_Rep2>::value)>::type> constexpr **duration** (const _Rep2 &__rep)
- template<typename _Rep2 , typename _Period2 , typename = typename enable_if<treat_as_floating_point<rep>::value || (ratio_↔ divide<_Period2, period>::den == 1 && !treat_as_floating_point<_Rep2>::value)>::type> constexpr **duration** (const [duration](#)<_Rep2, _Period2 > &__d)
- constexpr rep **count** () const
- template<typename _Rep2 = rep> enable_if<!treat_as_floating_point<_Rep2>::value, [duration](#) & >::type **operator%=(const rep &__rhs)**
- template<typename _Rep2 = rep> enable_if<!treat_as_floating_point<_Rep2>::value, [duration](#) & >::type **operator%=(const [duration](#) &__d)**
- [duration](#) & **operator*=(const rep &__rhs)**
- constexpr [duration](#) **operator+()** const
- [duration](#) & **operator++()**
- [duration](#) **operator++(int)**
- [duration](#) & **operator+=(const [duration](#) &__d)**
- constexpr [duration](#) **operator-()** const
- [duration](#) & **operator--()**
- [duration](#) **operator--(int)**
- [duration](#) & **operator-=(const [duration](#) &__d)**
- [duration](#) & **operator/=(const rep &__rhs)**
- [duration](#) & **operator=(const [duration](#) &)=default**

Static Public Member Functions

- static constexpr `duration` **max** ()
- static constexpr `duration` **min** ()
- static constexpr `duration` **zero** ()

5.648.1 Detailed Description

```
template<typename _Rep, typename _Period>
struct std::chrono::duration< _Rep, _Period >
```

`duration`

Definition at line 64 of file `chrono`.

The documentation for this struct was generated from the following file:

- `chrono`

5.649 `std::chrono::duration_values< _Rep >` Struct Template Reference

Static Public Member Functions

- static constexpr `_Rep` **max** ()
- static constexpr `_Rep` **min** ()
- static constexpr `_Rep` **zero** ()

5.649.1 Detailed Description

```
template<typename _Rep>
struct std::chrono::duration_values< _Rep >
```

`duration_values`

Definition at line 214 of file `chrono`.

The documentation for this struct was generated from the following file:

- `chrono`

5.650 `std::chrono::time_point< _Clock, _Dur >` Struct Template Reference

Public Types

- typedef `_Clock` **clock**
- typedef `_Dur` **duration**
- typedef `duration::period` **period**
- typedef `duration::rep` **rep**

Public Member Functions

- constexpr **time_point** (const duration &__dur)
- template<typename _Dur2 >
constexpr **time_point** (const [time_point](#)< clock, _Dur2 > &__t)
- [time_point](#) & **operator+=** (const duration &__dur)
- [time_point](#) & **operator-=** (const duration &__dur)
- constexpr duration **time_since_epoch** () const

Static Public Member Functions

- static constexpr [time_point](#) **max** ()
- static constexpr [time_point](#) **min** ()

5.650.1 Detailed Description

```
template<typename _Clock, typename _Dur>
struct std::chrono::time_point< _Clock, _Dur >
```

[time_point](#)

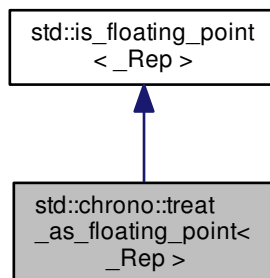
Definition at line 67 of file chrono.

The documentation for this struct was generated from the following file:

- [chrono](#)

5.651 std::chrono::treat_as_floating_point< _Rep > Struct Template Reference

Inheritance diagram for std::chrono::treat_as_floating_point< _Rep >:



5.651.1 Detailed Description

```
template<typename _Rep>
struct std::chrono::treat_as_floating_point<_Rep>
```

treat_as_floating_point

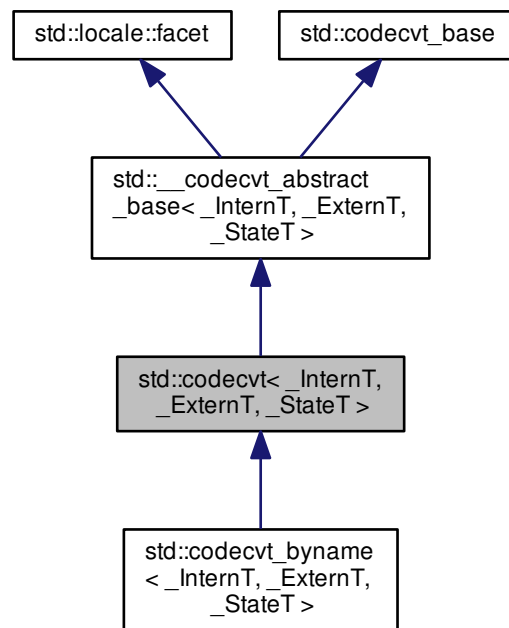
Definition at line 208 of file chrono.

The documentation for this struct was generated from the following file:

- [chrono](#)

5.652 std::codecvt<_InternT, _ExternT, _StateT > Class Template Reference

Inheritance diagram for std::codecvt<_InternT, _ExternT, _StateT >:



Public Types

- typedef `_ExternT` **extern_type**
- typedef `_InternT` **intern_type**
- typedef `codecvt_base::result` **result**
- typedef `_StateT` **state_type**

Public Member Functions

- **codecvt** (size_t __refs=0)
- **codecvt** (__c_locale __cloc, size_t __refs=0)
- bool **always_noconv** () const throw ()
- int **encoding** () const throw ()
- result **in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- int **length** (state_type &__state, const extern_type *__from, const extern_type *__end, size_t __max) const
- int **max_length** () const throw ()
- result **out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- result **unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const

Static Public Attributes

- static **locale::id** id

Protected Member Functions

- virtual bool **do_always_noconv** () const throw ()
- virtual int **do_encoding** () const throw ()
- virtual result **do_in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- virtual int **do_length** (state_type &, const extern_type *__from, const extern_type *__end, size_t __max) const
- virtual int **do_max_length** () const throw ()
- virtual result **do_out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- virtual result **do_unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const

Static Protected Member Functions

- static __c_locale **_S_clone_c_locale** (__c_locale &__cloc) throw ()
- static void **_S_create_c_locale** (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale &__cloc)
- static __c_locale **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static __c_locale **_S_lc_ctype_c_locale** (__c_locale __cloc, const char *__s)

Protected Attributes

- __c_locale **_M_c_locale_codecvt**

5.652.1 Detailed Description

```
template<typename _InternT, typename _ExternT, typename _StateT>
class std::codecvt<_InternT, _ExternT, _StateT >
```

Primary class template codecvt.

NB: Generic, mostly useless implementation.

Definition at line 274 of file codecvt.h.

5.652.2 Member Function Documentation

5.652.2.1 `template<typename _InternT, typename _ExternT, typename _StateT> virtual result std::codecvt<_InternT, _ExternT, _StateT>::do_out (state_type & __state, const intern_type * __from, const intern_type * __from_end, const intern_type * & __from_next, extern_type * __to, extern_type * __to_end, extern_type * & __to_next) const` [protected], [virtual]

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This function is a hook for derived classes to change the value returned.

See also

out for more information.

Implements [std::__codecvt_abstract_base<_InternT, _ExternT, _StateT >](#).

5.652.2.2 `template<typename _InternT, typename _ExternT, typename _StateT> result std::__codecvt_abstract_base<_InternT, _ExternT, _StateT>::in (state_type & __state, const extern_type * __from, const extern_type * __from_end, const extern_type * & __from_next, intern_type * __to, intern_type * __to_end, intern_type * & __to_next) const` [inline], [inherited]

Convert from external to internal character set.

Converts input string of extern_type to output string of intern_type. This is analogous to mbsrtowcs. It does this by calling codecvt::do_in.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in [from,from_end) are converted and written to [to,to_end). from_next and to_next are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, from_next and to_next are not affected.

The state argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how state is used.

The result returned is a member of codecvt_base::result. If all the input is converted, returns codecvt_base::ok. If no conversion is necessary, returns codecvt_base::noconv. If the input ends early or there is insufficient space in the output, returns codecvt_base::partial. Otherwise the conversion failed and codecvt_base::error is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

Definition at line 196 of file `codecvt.h`.

```
5.652.2.3 template<typename _InternT, typename _ExternT, typename _StateT> result std::__codecvt_abstract_base<
    _InternT, _ExternT, _StateT >::out ( state_type & __state, const intern_type * __from, const intern_type * __from_end,
    const intern_type * & __from_next, extern_type * __to, extern_type * __to_end, extern_type * & __to_next ) const
    [inline], [inherited]
```

Convert from internal to external character set.

Converts input string of `intern_type` to output string of `extern_type`. This is analogous to `wcsrtombs`. It does this by calling `codecvt::do_out`.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in `[from,from_end)` are converted and written to `[to,to_end)`. `from_next` and `to_next` are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, `from_next` and `to_next` are not affected.

The `state` argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how `state` is used.

The result returned is a member of `codecvt_base::result`. If all the input is converted, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the input ends early or there is insufficient space in the output, returns `codecvt_base::partial`. Otherwise the conversion failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

codecvt_base::result.

Definition at line 116 of file codecvt.h.

5.652.2.4 `template<typename _InternT, typename _ExternT, typename _StateT> result std::__codecvt_abstract_base<_InternT, _ExternT, _StateT>::unshift (state_type & __state, extern_type * __to, extern_type * __to_end, extern_type *& __to_next) const` `[inline], [inherited]`

Reset conversion state.

Writes characters to output that would restore *state* to initial conditions. The idea is that if a partial conversion occurs, then the converting the characters written by this function would leave the state in initial conditions, rather than partial conversion state. It does this by calling `codecvt::do_unshift()`.

For example, if 4 external characters always converted to 1 internal character, and input to `in()` had 6 external characters with state saved, this function would write two characters to the output and set the state to initialized conditions.

The source and destination character sets are determined by the facet's locale, internal and external types.

The result returned is a member of `codecvt_base::result`. If the state could be reset and data written, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the output has insufficient space, returns `codecvt_base::partial`. Otherwise the reset failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

codecvt_base::result.

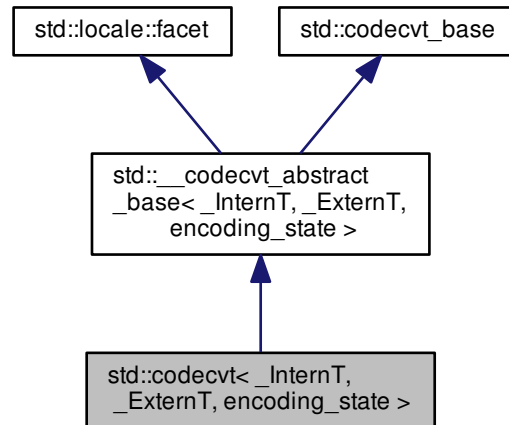
Definition at line 155 of file codecvt.h.

The documentation for this class was generated from the following file:

- [codecvt.h](#)

5.653 `std::codecvt< _InternT, _ExternT, encoding_state >` Class Template Reference

Inheritance diagram for `std::codecvt< _InternT, _ExternT, encoding_state >`:



Public Types

- typedef `state_type::descriptor_type` **descriptor_type**
- typedef `_ExternT` **extern_type**
- typedef `_InternT` **intern_type**
- typedef `codecvt_base::result` **result**
- typedef `__gnu_cxx::encoding_state` **state_type**

Public Member Functions

- **codecvt** (`size_t __refs=0`)
- **codecvt** (`state_type &__enc, size_t __refs=0`)
- **bool always_noconv** () const throw ()
- **int encoding** () const throw ()
- **result in** (`state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__from_next, intern_type *__to, intern_type *__to_end, intern_type *__to_next`) const
- **int length** (`state_type &__state, const extern_type *__from, const extern_type *__end, size_t __max`) const
- **int max_length** () const throw ()
- **result out** (`state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__from_next, extern_type *__to, extern_type *__to_end, extern_type *__to_next`) const
- **result unshift** (`state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__to_next`) const

Static Public Attributes

- static `locale::id` **id**

Protected Member Functions

- virtual bool **do_always_noconv** () const throw ()
- virtual int **do_encoding** () const throw ()
- virtual result **do_in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *&__from_next, intern_type *__to, intern_type *__to_end, intern_type *&__to_next) const
- virtual int **do_length** (state_type &, const extern_type *__from, const extern_type *__end, size_t __max) const
- virtual int **do_max_length** () const throw ()
- virtual result **do_out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *&__from_next, extern_type *__to, extern_type *__to_end, extern_type *&__to_next) const
- virtual result **do_unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *&__to_next) const

Static Protected Member Functions

- static __c_locale **_S_clone_c_locale** (__c_locale &__cloc) throw ()
- static void **_S_create_c_locale** (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale &__cloc)
- static __c_locale **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static __c_locale **_S_lc_ctype_c_locale** (__c_locale __cloc, const char *__s)

5.653.1 Detailed Description

```
template<typename _InternT, typename _ExternT>
class std::codecvt<_InternT, _ExternT, encoding_state >
```

codecvt<InternT, _ExternT, encoding_state> specialization.

Definition at line 233 of file codecvt_specializations.h.

5.653.2 Member Function Documentation

5.653.2.1 `template<typename _InternT, typename _ExternT > codecvt_base::result std::codecvt<_InternT, _ExternT, encoding_state >::do_out (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *&__from_next, extern_type *__to, extern_type *__to_end, extern_type *&__to_next) const` [protected], [virtual]

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This function is a hook for derived classes to change the value returned.

See also

out for more information.

Implements `std::__codecvt_abstract_base<_InternT, _ExternT, encoding_state >`.

Definition at line 309 of file codecvt_specializations.h.

References `std::min()`.

```
5.653.2.2 result std::__codecvt_abstract_base<_InternT, _ExternT, encoding_state >::in ( state_type & __state,
const extern_type * __from, const extern_type * __from_end, const extern_type * & __from_next, intern_type * __to,
intern_type * __to_end, intern_type * & __to_next ) const [inline],[inherited]
```

Convert from external to internal character set.

Converts input string of `extern_type` to output string of `intern_type`. This is analogous to `mbsrtowcs`. It does this by calling `codecvt::do_in`.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in `[from,from_end)` are converted and written to `[to,to_end)`. `from_next` and `to_next` are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, `from_next` and `to_next` are not affected.

The `state` argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how `state` is used.

The result returned is a member of `codecvt_base::result`. If all the input is converted, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the input ends early or there is insufficient space in the output, returns `codecvt_base::partial`. Otherwise the conversion failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

Definition at line 196 of file `codecvt.h`.

```
5.653.2.3 result std::__codecvt_abstract_base<_InternT, _ExternT, encoding_state >::out ( state_type & __state,
const intern_type * __from, const intern_type * __from_end, const intern_type * & __from_next, extern_type * __to,
extern_type * __to_end, extern_type * & __to_next ) const [inline],[inherited]
```

Convert from internal to external character set.

Converts input string of `intern_type` to output string of `extern_type`. This is analogous to `wcsrtombs`. It does this by calling `codecvt::do_out`.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in `[from,from_end)` are converted and written to `[to,to_end)`. `from_next` and `to_next` are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, `from_next` and `to_next` are not affected.

The *state* argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how *state* is used.

The result returned is a member of `codecvt_base::result`. If all the input is converted, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the input ends early or there is insufficient space in the output, returns `codecvt_base::partial`. Otherwise the conversion failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

Definition at line 116 of file `codecvt.h`.

5.653.2.4 `result std::__codecvt_abstract_base<_InternT, _ExternT, encoding_state >::unshift (state_type & __state, extern_type * __to, extern_type * __to_end, extern_type *& __to_next) const` [inline],[inherited]

Reset conversion state.

Writes characters to output that would restore *state* to initial conditions. The idea is that if a partial conversion occurs, then the converting the characters written by this function would leave the state in initial conditions, rather than partial conversion state. It does this by calling `codecvt::do_unshift()`.

For example, if 4 external characters always converted to 1 internal character, and input to `in()` had 6 external characters with state saved, this function would write two characters to the output and set the state to initialized conditions.

The source and destination character sets are determined by the facet's locale, internal and external types.

The result returned is a member of `codecvt_base::result`. If the state could be reset and data written, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the output has insufficient space, returns `codecvt_base::partial`. Otherwise the reset failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

codecvt_base::result.

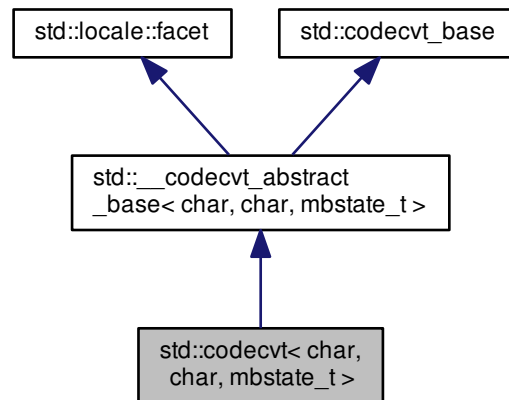
Definition at line 155 of file codecvt.h.

The documentation for this class was generated from the following file:

- [codecvt_specializations.h](#)

5.654 std::codecvt< char, char, mbstate_t > Class Template Reference

Inheritance diagram for std::codecvt< char, char, mbstate_t >:



Public Types

- typedef char **extern_type**
- typedef char **intern_type**
- typedef codecvt_base::result **result**
- typedef mbstate_t **state_type**

Public Member Functions

- **codecvt** (size_t __refs=0)
- **codecvt** (__c_locale __cloc, size_t __refs=0)
- bool **always_noconv** () const throw ()
- int **encoding** () const throw ()
- result **in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- int **length** (state_type &__state, const extern_type *__from, const extern_type *__end, size_t __max) const
- int **max_length** () const throw ()
- result **out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- result **unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const

Static Public Attributes

- static [locale::id](#) id

Protected Member Functions

- virtual bool **do_always_noconv** () const throw ()
- virtual int **do_encoding** () const throw ()
- virtual result **do_in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- virtual int **do_length** (state_type &, const extern_type *__from, const extern_type *__end, size_t __max) const
- virtual int **do_max_length** () const throw ()
- virtual result **do_out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- virtual result **do_unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const

Static Protected Member Functions

- static __c_locale **_S_clone_c_locale** (__c_locale &__cloc) throw ()
- static void **_S_create_c_locale** (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale &__cloc)
- static __c_locale **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static __c_locale **_S_lc_ctype_c_locale** (__c_locale __cloc, const char *__s)

Protected Attributes

- __c_locale **_M_c_locale_codecvt**

Friends

- class **messages< char >**

5.654.1 Detailed Description

template<>

class std::codecvt< char, char, mbstate_t >

class codecvt<char, char, mbstate_t> specialization.

Definition at line 338 of file codecvt.h.

5.654.2 Member Function Documentation

5.654.2.1 virtual result std::codecvt< char, char, mbstate_t >::do_out (state_type & __state, const intern_type * __from, const intern_type * __from_end, const intern_type * & __from_next, extern_type * __to, extern_type * __to_end, extern_type * & __to_next) const [protected], [virtual]

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This function is a hook for derived classes to change the value returned.

See also

out for more information.

Implements [std::__codecvt_abstract_base< char, char, mbstate_t >](#).

5.654.2.2 result std::__codecvt_abstract_base< char, char, mbstate_t >::in (state_type & __state, const extern_type * __from, const extern_type * __from_end, const extern_type * & __from_next, intern_type * __to, intern_type * __to_end, intern_type * & __to_next) const [inline], [inherited]

Convert from external to internal character set.

Converts input string of extern_type to output string of intern_type. This is analogous to mbsrtowcs. It does this by calling codecvt::do_in.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in [from,from_end) are converted and written to [to,to_end). from_next and to_next are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, from_next and to_next are not affected.

The *state* argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how *state* is used.

The result returned is a member of codecvt_base::result. If all the input is converted, returns codecvt_base::ok. If no conversion is necessary, returns codecvt_base::noconv. If the input ends early or there is insufficient space in the output, returns codecvt_base::partial. Otherwise the conversion failed and codecvt_base::error is returned.

Parameters

<i>__state</i>	Persistent conversion state data.
<i>__from</i>	Start of input.
<i>__from_end</i>	End of input.
<i>__from_next</i>	Returns start of unconverted data.
<i>__to</i>	Start of output buffer.
<i>__to_end</i>	End of output buffer.
<i>__to_next</i>	Returns start of unused output area.

Returns

codecvt_base::result.

Definition at line 196 of file codecvt.h.

5.654.2.3 **result** std::__codecvt_abstract_base< char , char , mbstate_t >::out (state_type & __state, const intern_type * __from, const intern_type * __from_end, const intern_type *& __from_next, extern_type * __to, extern_type * __to_end, extern_type *& __to_next) const [inline],[inherited]

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This is analogous to wcsrtombs. It does this by calling codecvt::do_out.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in [from,from_end) are converted and written to [to,to_end). from_next and to_next are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, from_next and to_next are not affected.

The *state* argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how *state* is used.

The result returned is a member of codecvt_base::result. If all the input is converted, returns codecvt_base::ok. If no conversion is necessary, returns codecvt_base::noconv. If the input ends early or there is insufficient space in the output, returns codecvt_base::partial. Otherwise the conversion failed and codecvt_base::error is returned.

Parameters

__state	Persistent conversion state data.
__from	Start of input.
__from_end	End of input.
__from_next	Returns start of unconverted data.
__to	Start of output buffer.
__to_end	End of output buffer.
__to_next	Returns start of unused output area.

Returns

codecvt_base::result.

Definition at line 116 of file codecvt.h.

5.654.2.4 **result** std::__codecvt_abstract_base< char , char , mbstate_t >::unshift (state_type & __state, extern_type * __to, extern_type * __to_end, extern_type *& __to_next) const [inline],[inherited]

Reset conversion state.

Writes characters to output that would restore *state* to initial conditions. The idea is that if a partial conversion occurs, then the converting the characters written by this function would leave the state in initial conditions, rather than partial conversion state. It does this by calling `codecv_t::do_unshift()`.

For example, if 4 external characters always converted to 1 internal character, and input to `in()` had 6 external characters with state saved, this function would write two characters to the output and set the state to initialized conditions.

The source and destination character sets are determined by the facet's locale, internal and external types.

The result returned is a member of `codecv_t_base::result`. If the state could be reset and data written, returns `codecv_t_base::ok`. If no conversion is necessary, returns `codecv_t_base::noconv`. If the output has insufficient space, returns `codecv_t_base::partial`. Otherwise the reset failed and `codecv_t_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecv_t_base::result`.

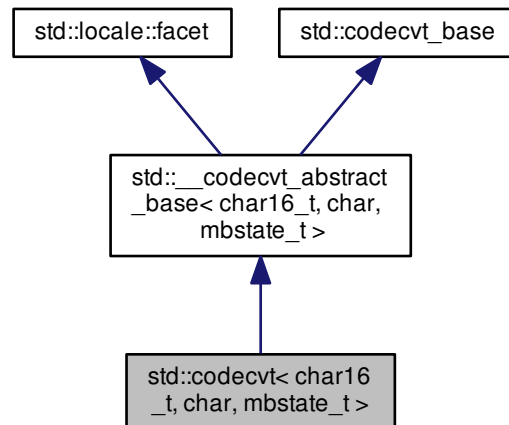
Definition at line 155 of file `codecv_t.h`.

The documentation for this class was generated from the following file:

- [codecv_t.h](#)

5.655 std::codecvt< char16_t, char, mbstate_t > Class Template Reference

Inheritance diagram for std::codecvt< char16_t, char, mbstate_t >:



Public Types

- typedef char **extern_type**
- typedef char16_t **intern_type**
- typedef codecvt_base::result **result**
- typedef mbstate_t **state_type**

Public Member Functions

- **codecvt** (size_t __refs=0)
- bool **always_noconv** () const throw ()
- int **encoding** () const throw ()
- result **in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- int **length** (state_type &__state, const extern_type *__from, const extern_type *__end, size_t __max) const
- int **max_length** () const throw ()
- result **out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- result **unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const

Static Public Attributes

- static `locale::id` **id**

Protected Member Functions

- virtual bool **do_always_noconv** () const throw ()
- virtual int **do_encoding** () const throw ()
- virtual result **do_in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__from_next, intern_type *__to, intern_type *__to_end, intern_type *__to_next) const
- virtual int **do_length** (state_type &, const extern_type *__from, const extern_type *__end, size_t __max) const
- virtual int **do_max_length** () const throw ()
- virtual result **do_out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__from_next, extern_type *__to, extern_type *__to_end, extern_type *__to_next) const
- virtual result **do_unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__to_next) const

Static Protected Member Functions

- static __c_locale **_S_clone_c_locale** (__c_locale &__cloc) throw ()
- static void **_S_create_c_locale** (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale &__cloc)
- static __c_locale **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static __c_locale **_S_lc_ctype_c_locale** (__c_locale __cloc, const char *__s)

5.655.1 Detailed Description

```
template<>
class std::codecvt< char16_t, char, mbstate_t >
```

Class codecvt<char16_t, char, mbstate_t> specialization.

Converts between UTF-16 and UTF-8.

Definition at line 468 of file codecvt.h.

5.655.2 Member Function Documentation

5.655.2.1 virtual result std::codecvt< char16_t, char, mbstate_t >::do_out (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__from_next, extern_type *__to, extern_type *__to_end, extern_type *__to_next) const [protected],[virtual]

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This function is a hook for derived classes to change the value returned.

See also

out for more information.

Implements [std::__codecvt_abstract_base< char16_t, char, mbstate_t >](#).

5.655.2.2 **result** std::__codecvt_abstract_base< char16_t, char, mbstate_t >::in (state_type & __state, const extern_type * __from, const extern_type * __from_end, const extern_type * & __from_next, intern_type * __to, intern_type * __to_end, intern_type * & __to_next) const [inline], [inherited]

Convert from external to internal character set.

Converts input string of extern_type to output string of intern_type. This is analogous to mbsrtowcs. It does this by calling codecvt::do_in.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in [from,from_end) are converted and written to [to,to_end). from_next and to_next are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, from_next and to_next are not affected.

The state argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how state is used.

The result returned is a member of codecvt_base::result. If all the input is converted, returns codecvt_base::ok. If no conversion is necessary, returns codecvt_base::noconv. If the input ends early or there is insufficient space in the output, returns codecvt_base::partial. Otherwise the conversion failed and codecvt_base::error is returned.

Parameters

__state	Persistent conversion state data.
__from	Start of input.
__from_end	End of input.
__from_next	Returns start of unconverted data.
__to	Start of output buffer.
__to_end	End of output buffer.
__to_next	Returns start of unused output area.

Returns

codecvt_base::result.

Definition at line 196 of file codecvt.h.

5.655.2.3 **result** std::__codecvt_abstract_base< char16_t, char, mbstate_t >::out (state_type & __state, const intern_type * __from, const intern_type * __from_end, const intern_type * & __from_next, extern_type * __to, extern_type * __to_end, extern_type * & __to_next) const [inline], [inherited]

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This is analogous to wcsrtombs. It does this by calling codecvt::do_out.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in `[from,from_end)` are converted and written to `[to,to_end)`. `from_next` and `to_next` are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, `from_next` and `to_next` are not affected.

The *state* argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how *state* is used.

The result returned is a member of `codecvt_base::result`. If all the input is converted, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the input ends early or there is insufficient space in the output, returns `codecvt_base::partial`. Otherwise the conversion failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

Definition at line 116 of file `codecvt.h`.

5.655.2.4 `result std::__codecvt_abstract_base< char16_t, char, mbstate_t >::unshift (state_type & __state, extern_type * __to, extern_type * __to_end, extern_type *& __to_next) const` `[inline],[inherited]`

Reset conversion state.

Writes characters to output that would restore *state* to initial conditions. The idea is that if a partial conversion occurs, then the converting the characters written by this function would leave the state in initial conditions, rather than partial conversion state. It does this by calling `codecvt::do_unshift()`.

For example, if 4 external characters always converted to 1 internal character, and input to `in()` had 6 external characters with state saved, this function would write two characters to the output and set the state to initialized conditions.

The source and destination character sets are determined by the facet's locale, internal and external types.

The result returned is a member of `codecvt_base::result`. If the state could be reset and data written, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the output has insufficient space, returns `codecvt_base::partial`. Otherwise the reset failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

codecvt_base::result.

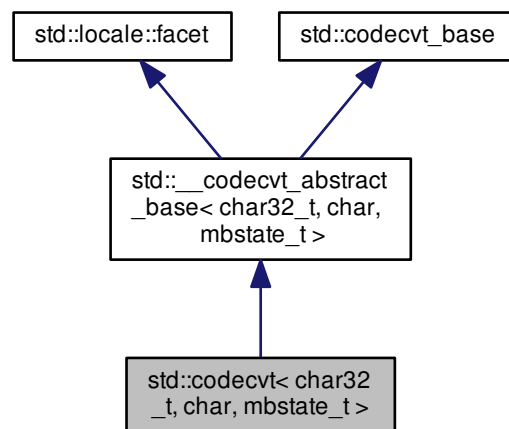
Definition at line 155 of file codecvt.h.

The documentation for this class was generated from the following file:

- [codecvt.h](#)

5.656 std::codecvt< char32_t, char, mbstate_t > Class Template Reference

Inheritance diagram for std::codecvt< char32_t, char, mbstate_t >:



Public Types

- typedef char **extern_type**
- typedef char32_t **intern_type**
- typedef codecvt_base::result **result**
- typedef mbstate_t **state_type**

Public Member Functions

- **codecvt** (size_t __refs=0)
- bool **always_noconv** () const throw ()
- int **encoding** () const throw ()
- result **in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- int **length** (state_type &__state, const extern_type *__from, const extern_type *__end, size_t __max) const
- int **max_length** () const throw ()
- result **out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- result **unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const

Static Public Attributes

- static [locale::id](#) `id`

Protected Member Functions

- virtual bool **do_always_noconv** () const throw ()
- virtual int **do_encoding** () const throw ()
- virtual result **do_in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__from_next, intern_type *__to, intern_type *__to_end, intern_type *__to_next) const
- virtual int **do_length** (state_type &, const extern_type *__from, const extern_type *__end, size_t __max) const
- virtual int **do_max_length** () const throw ()
- virtual result **do_out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__from_next, extern_type *__to, extern_type *__to_end, extern_type *__to_next) const
- virtual result **do_unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__to_next) const

Static Protected Member Functions

- static __c_locale **S_clone_c_locale** (__c_locale &__cloc) throw ()
- static void **S_create_c_locale** (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void **S_destroy_c_locale** (__c_locale &__cloc)
- static __c_locale **S_get_c_locale** ()
- static const char * **S_get_c_name** () throw ()
- static __c_locale **S_lc_ctype_c_locale** (__c_locale __cloc, const char *__s)

5.656.1 Detailed Description

```
template<>
class std::codecvt< char32_t, char, mbstate_t >
```

Class `codecvt<char32_t, char, mbstate_t>` specialization.

Converts between UTF-32 and UTF-8.

Definition at line 525 of file `codecvt.h`.

5.656.2 Member Function Documentation

5.656.2.1 virtual result `std::codecvt< char32_t, char, mbstate_t >::do_out (state_type & __state, const intern_type * __from, const intern_type * __from_end, const intern_type * __from_next, extern_type * __to, extern_type * __to_end, extern_type * __to_next) const` `[protected]`, `[virtual]`

Convert from internal to external character set.

Converts input string of `intern_type` to output string of `extern_type`. This function is a hook for derived classes to change the value returned.

See also

`do_out` for more information.

Implements `std::__codecvt_abstract_base< char32_t, char, mbstate_t >`.

5.656.2.2 **result** std::__codecvt_abstract_base< char32_t, char, mbstate_t >::in (state_type & __state, const extern_type * __from, const extern_type * __from_end, const extern_type *& __from_next, intern_type * __to, intern_type * __to_end, intern_type *& __to_next) const [inline],[inherited]

Convert from external to internal character set.

Converts input string of extern_type to output string of intern_type. This is analogous to mbsrtowcs. It does this by calling codecvt::do_in.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in [from,from_end) are converted and written to [to,to_end). from_next and to_next are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, from_next and to_next are not affected.

The state argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how state is used.

The result returned is a member of codecvt_base::result. If all the input is converted, returns codecvt_base::ok. If no conversion is necessary, returns codecvt_base::noconv. If the input ends early or there is insufficient space in the output, returns codecvt_base::partial. Otherwise the conversion failed and codecvt_base::error is returned.

Parameters

__state	Persistent conversion state data.
__from	Start of input.
__from_end	End of input.
__from_next	Returns start of unconverted data.
__to	Start of output buffer.
__to_end	End of output buffer.
__to_next	Returns start of unused output area.

Returns

codecvt_base::result.

Definition at line 196 of file codecvt.h.

5.656.2.3 **result** std::__codecvt_abstract_base< char32_t, char, mbstate_t >::out (state_type & __state, const intern_type * __from, const intern_type * __from_end, const intern_type *& __from_next, extern_type * __to, extern_type * __to_end, extern_type *& __to_next) const [inline],[inherited]

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This is analogous to wcsrtombs. It does this by calling codecvt::do_out.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in `[from,from_end)` are converted and written to `[to,to_end)`. `from_next` and `to_next` are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, `from_next` and `to_next` are not affected.

The *state* argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how *state* is used.

The result returned is a member of `codecvt_base::result`. If all the input is converted, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the input ends early or there is insufficient space in the output, returns `codecvt_base::partial`. Otherwise the conversion failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

Definition at line 116 of file `codecvt.h`.

5.656.2.4 `result std::__codecvt_abstract_base< char32_t, char, mbstate_t >::unshift (state_type & __state, extern_type * __to, extern_type * __to_end, extern_type *& __to_next) const` `[inline],[inherited]`

Reset conversion state.

Writes characters to output that would restore *state* to initial conditions. The idea is that if a partial conversion occurs, then the converting the characters written by this function would leave the state in initial conditions, rather than partial conversion state. It does this by calling `codecvt::do_unshift()`.

For example, if 4 external characters always converted to 1 internal character, and input to `in()` had 6 external characters with state saved, this function would write two characters to the output and set the state to initialized conditions.

The source and destination character sets are determined by the facet's locale, internal and external types.

The result returned is a member of `codecvt_base::result`. If the state could be reset and data written, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the output has insufficient space, returns `codecvt_base::partial`. Otherwise the reset failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

codecvt_base::result.

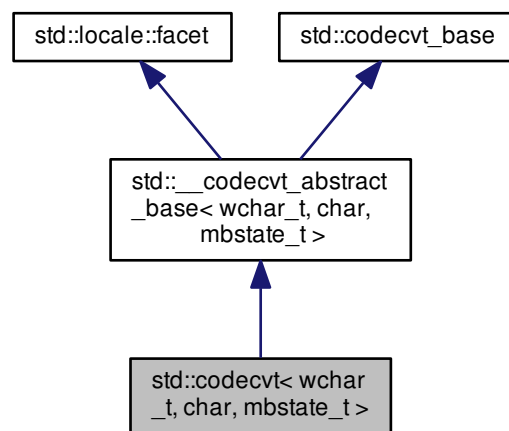
Definition at line 155 of file codecvt.h.

The documentation for this class was generated from the following file:

- [codecvt.h](#)

5.657 std::codecvt< wchar_t, char, mbstate_t > Class Template Reference

Inheritance diagram for std::codecvt< wchar_t, char, mbstate_t >:



Public Types

- typedef char **extern_type**
- typedef wchar_t **intern_type**
- typedef codecvt_base::result **result**
- typedef mbstate_t **state_type**

Public Member Functions

- **codecvt** (size_t __refs=0)
- **codecvt** (__c_locale __cloc, size_t __refs=0)
- bool **always_noconv** () const throw ()
- int **encoding** () const throw ()
- result **in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- int **length** (state_type &__state, const extern_type *__from, const extern_type *__end, size_t __max) const
- int **max_length** () const throw ()
- result **out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- result **unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const

Static Public Attributes

- static `locale::id` `id`

Protected Member Functions

- virtual bool **do_always_noconv** () const throw ()
- virtual int **do_encoding** () const throw ()
- virtual result **do_in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- virtual int **do_length** (state_type &, const extern_type *__from, const extern_type *__end, size_t __max) const
- virtual int **do_max_length** () const throw ()
- virtual result **do_out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- virtual result **do_unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to←__next) const

Static Protected Member Functions

- static __c_locale **_S_clone_c_locale** (__c_locale &__cloc) throw ()
- static void **_S_create_c_locale** (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale &__cloc)
- static __c_locale **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static __c_locale **_S_lc_ctype_c_locale** (__c_locale __cloc, const char *__s)

Protected Attributes

- __c_locale **_M_c_locale_codecvt**

Friends

- class **messages**< `wchar_t` >

5.657.1 Detailed Description

```
template<>
class std::codecvt< wchar_t, char, mbstate_t >
```

Class `codecvt<wchar_t, char, mbstate_t>` specialization.

Converts between narrow and wide characters in the native character set

Definition at line 401 of file `codecvt.h`.

5.657.2 Member Function Documentation

5.657.2.1 virtual result std::codecvt< wchar_t, char, mbstate_t >::do_out (state_type & __state, const intern_type * __from, const intern_type * __from_end, const intern_type * & __from_next, extern_type * __to, extern_type * __to_end, extern_type * & __to_next) const [protected], [virtual]

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This function is a hook for derived classes to change the value returned.

See also

out for more information.

Implements [std::__codecvt_abstract_base< wchar_t, char, mbstate_t >](#).

5.657.2.2 result std::__codecvt_abstract_base< wchar_t, char, mbstate_t >::in (state_type & __state, const extern_type * __from, const extern_type * __from_end, const extern_type * & __from_next, intern_type * __to, intern_type * __to_end, intern_type * & __to_next) const [inline], [inherited]

Convert from external to internal character set.

Converts input string of extern_type to output string of intern_type. This is analogous to mbsrtowcs. It does this by calling codecvt::do_in.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in [from,from_end) are converted and written to [to,to_end). from_next and to_next are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, from_next and to_next are not affected.

The *state* argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how *state* is used.

The result returned is a member of codecvt_base::result. If all the input is converted, returns codecvt_base::ok. If no conversion is necessary, returns codecvt_base::noconv. If the input ends early or there is insufficient space in the output, returns codecvt_base::partial. Otherwise the conversion failed and codecvt_base::error is returned.

Parameters

<i>__state</i>	Persistent conversion state data.
<i>__from</i>	Start of input.
<i>__from_end</i>	End of input.
<i>__from_next</i>	Returns start of unconverted data.
<i>__to</i>	Start of output buffer.
<i>__to_end</i>	End of output buffer.
<i>__to_next</i>	Returns start of unused output area.

Returns

codecvt_base::result.

Definition at line 196 of file codecvt.h.

5.657.2.3 **result** std::__codecvt_abstract_base< wchar_t, char, mbstate_t >::out (state_type & __state, const intern_type * __from, const intern_type * __from_end, const intern_type *& __from_next, extern_type * __to, extern_type * __to_end, extern_type *& __to_next) const [inline],[inherited]

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This is analogous to wcsrtombs. It does this by calling codecvt::do_out.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in [from,from_end) are converted and written to [to,to_end). from_next and to_next are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, from_next and to_next are not affected.

The *state* argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how *state* is used.

The result returned is a member of codecvt_base::result. If all the input is converted, returns codecvt_base::ok. If no conversion is necessary, returns codecvt_base::noconv. If the input ends early or there is insufficient space in the output, returns codecvt_base::partial. Otherwise the conversion failed and codecvt_base::error is returned.

Parameters

__state	Persistent conversion state data.
__from	Start of input.
__from_end	End of input.
__from_next	Returns start of unconverted data.
__to	Start of output buffer.
__to_end	End of output buffer.
__to_next	Returns start of unused output area.

Returns

codecvt_base::result.

Definition at line 116 of file codecvt.h.

5.657.2.4 **result** std::__codecvt_abstract_base< wchar_t, char, mbstate_t >::unshift (state_type & __state, extern_type * __to, extern_type * __to_end, extern_type *& __to_next) const [inline],[inherited]

Reset conversion state.

Writes characters to output that would restore *state* to initial conditions. The idea is that if a partial conversion occurs, then the converting the characters written by this function would leave the state in initial conditions, rather than partial conversion state. It does this by calling `codecvt::do_unshift()`.

For example, if 4 external characters always converted to 1 internal character, and input to `in()` had 6 external characters with state saved, this function would write two characters to the output and set the state to initialized conditions.

The source and destination character sets are determined by the facet's locale, internal and external types.

The result returned is a member of `codecvt_base::result`. If the state could be reset and data written, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the output has insufficient space, returns `codecvt_base::partial`. Otherwise the reset failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

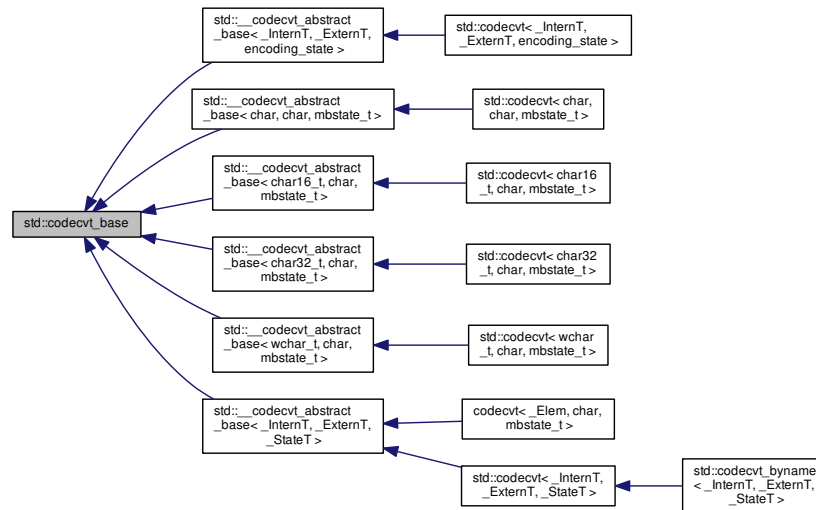
Definition at line 155 of file `codecvt.h`.

The documentation for this class was generated from the following file:

- [codecvt.h](#)

5.658 std::codecvt_base Class Reference

Inheritance diagram for std::codecvt_base:



Public Types

- enum **result** { **ok**, **partial**, **error**, **noconv** }

5.658.1 Detailed Description

Empty base class for codecvt facet [22.2.1.5].

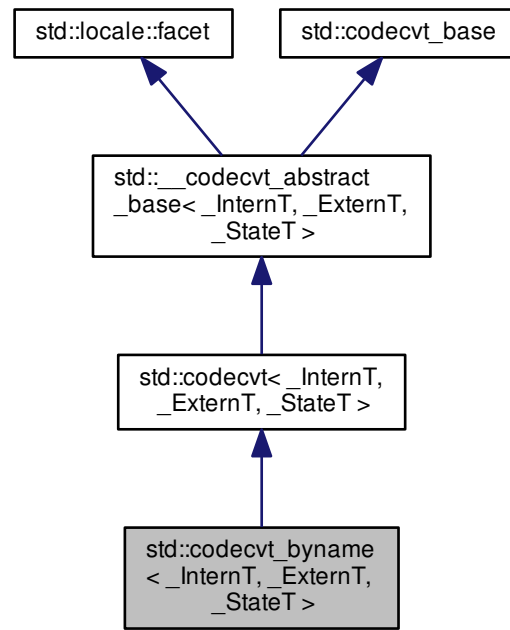
Definition at line 46 of file codecvt.h.

The documentation for this class was generated from the following file:

- [codecvt.h](#)

5.659 std::codecvt_byname< _InternT, _ExternT, _StateT > Class Template Reference

Inheritance diagram for std::codecvt_byname< _InternT, _ExternT, _StateT >:



Public Types

- typedef `_ExternT` **extern_type**
- typedef `_InternT` **intern_type**
- typedef `codecvt_base::result` **result**
- typedef `_StateT` **state_type**

Public Member Functions

- **codecvt_byname** (const char *__s, size_t __refs=0)
- **codecvt_byname** (const [string](#) &__s, size_t __refs=0)
- bool **always_noconv** () const throw ()
- int **encoding** () const throw ()
- result **in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *__&__from_next, intern_type *__to, intern_type *__to_end, intern_type *__&__to_next) const
- int **length** (state_type &__state, const extern_type *__from, const extern_type *__end, size_t __max) const
- int **max_length** () const throw ()
- result **out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *__&__from_next, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const
- result **unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *__&__to_next) const

Static Public Attributes

- static `locale::id`

Protected Member Functions

- virtual bool **do_always_noconv** () const throw ()
- virtual int **do_encoding** () const throw ()
- virtual result **do_in** (state_type &__state, const extern_type *__from, const extern_type *__from_end, const extern_type *&__from_next, intern_type *__to, intern_type *__to_end, intern_type *&__to_next) const
- virtual int **do_length** (state_type &, const extern_type *__from, const extern_type *__end, size_t __max) const
- virtual int **do_max_length** () const throw ()
- virtual result **do_out** (state_type &__state, const intern_type *__from, const intern_type *__from_end, const intern_type *&__from_next, extern_type *__to, extern_type *__to_end, extern_type *&__to_next) const
- virtual result **do_unshift** (state_type &__state, extern_type *__to, extern_type *__to_end, extern_type *&__to←__next) const

Static Protected Member Functions

- static `__c_locale _S_clone_c_locale` (__c_locale &__cloc) throw ()
- static void **_S_create_c_locale** (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale &__cloc)
- static `__c_locale _S_get_c_locale` ()
- static const char * **_S_get_c_name** () throw ()
- static `__c_locale _S_lc_ctype_c_locale` (__c_locale __cloc, const char *__s)

Protected Attributes

- `__c_locale _M_c_locale_codecvt`

5.659.1 Detailed Description

```
template<typename _InternT, typename _ExternT, typename _StateT>
class std::codecvt_byname< _InternT, _ExternT, _StateT >
```

class codecvt_byname [22.2.1.6].

Definition at line 582 of file codecvt.h.

5.659.2 Member Function Documentation

5.659.2.1 `template<typename _InternT, typename _ExternT, typename _StateT> virtual result std::codecvt< _InternT, _ExternT, _StateT >::do_out (state_type & __state, const intern_type * __from, const intern_type * __from_end, const intern_type * & __from_next, extern_type * __to, extern_type * __to_end, extern_type * & __to_next) const` `[protected]`, `[virtual]`, `[inherited]`

Convert from internal to external character set.

Converts input string of `intern_type` to output string of `extern_type`. This function is a hook for derived classes to change the value returned.

See also

`out` for more information.

Implements [std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >](#).

5.659.2.2 `template<typename _InternT, typename _ExternT, typename _StateT> result std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >::in (state_type & __state, const extern_type * __from, const extern_type * __from_end, const extern_type * & __from_next, intern_type * __to, intern_type * __to_end, intern_type * & __to_next) const` `[inline]`, `[inherited]`

Convert from external to internal character set.

Converts input string of `extern_type` to output string of `intern_type`. This is analogous to `mbsrtowcs`. It does this by calling `codecvt::do_in`.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in `[from,from_end)` are converted and written to `[to,to_end)`. `from_next` and `to_next` are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, `from_next` and `to_next` are not affected.

The `state` argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how `state` is used.

The result returned is a member of `codecvt_base::result`. If all the input is converted, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the input ends early or there is insufficient space in the output, returns `codecvt_base::partial`. Otherwise the conversion failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__from</code>	Start of input.
<code>__from_end</code>	End of input.
<code>__from_next</code>	Returns start of unconverted data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

codecvt_base::result.

Definition at line 196 of file codecvt.h.

```
5.659.2.3 template<typename _InternT, typename _ExternT, typename _StateT> result std::__codecvt_abstract_base<
    _InternT, _ExternT, _StateT>::out ( state_type & __state, const intern_type * __from, const intern_type * __from_end,
    const intern_type * & __from_next, extern_type * __to, extern_type * __to_end, extern_type * & __to_next ) const
    [inline], [inherited]
```

Convert from internal to external character set.

Converts input string of intern_type to output string of extern_type. This is analogous to wcsrtombs. It does this by calling codecvt::do_out.

The source and destination character sets are determined by the facet's locale, internal and external types.

The characters in [from,from_end) are converted and written to [to,to_end). from_next and to_next are set to point to the character following the last successfully converted character, respectively. If the result needed no conversion, from_next and to_next are not affected.

The *state* argument should be initialized if the input is at the beginning and carried from a previous call if continuing conversion. There are no guarantees about how *state* is used.

The result returned is a member of codecvt_base::result. If all the input is converted, returns codecvt_base::ok. If no conversion is necessary, returns codecvt_base::noconv. If the input ends early or there is insufficient space in the output, returns codecvt_base::partial. Otherwise the conversion failed and codecvt_base::error is returned.

Parameters

__state	Persistent conversion state data.
__from	Start of input.
__from_end	End of input.
__from_next	Returns start of unconverted data.
__to	Start of output buffer.
__to_end	End of output buffer.
__to_next	Returns start of unused output area.

Returns

codecvt_base::result.

Definition at line 116 of file codecvt.h.

```
5.659.2.4 template<typename _InternT, typename _ExternT, typename _StateT> result std::__codecvt_abstract_base<
    _InternT, _ExternT, _StateT>::unshift ( state_type & __state, extern_type * __to, extern_type * __to_end, extern_type
    * & __to_next ) const [inline], [inherited]
```

Reset conversion state.

Writes characters to output that would restore *state* to initial conditions. The idea is that if a partial conversion occurs, then the converting the characters written by this function would leave the state in initial conditions, rather than partial conversion state. It does this by calling `codecvt::do_unshift()`.

For example, if 4 external characters always converted to 1 internal character, and input to `in()` had 6 external characters with state saved, this function would write two characters to the output and set the state to initialized conditions.

The source and destination character sets are determined by the facet's locale, internal and external types.

The result returned is a member of `codecvt_base::result`. If the state could be reset and data written, returns `codecvt_base::ok`. If no conversion is necessary, returns `codecvt_base::noconv`. If the output has insufficient space, returns `codecvt_base::partial`. Otherwise the reset failed and `codecvt_base::error` is returned.

Parameters

<code>__state</code>	Persistent conversion state data.
<code>__to</code>	Start of output buffer.
<code>__to_end</code>	End of output buffer.
<code>__to_next</code>	Returns start of unused output area.

Returns

`codecvt_base::result`.

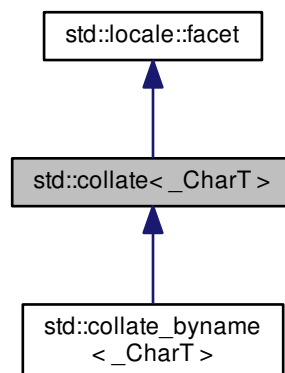
Definition at line 155 of file `codecvt.h`.

The documentation for this class was generated from the following file:

- [codecvt.h](#)

5.660 `std::collate<_CharT>` Class Template Reference

Inheritance diagram for `std::collate<_CharT>`:



Public Types

- typedef `_CharT` [char_type](#)
- typedef [basic_string](#)< `_CharT` > [string_type](#)

Public Member Functions

- [collate](#) (size_t __refs=0)
- [collate](#) (__c_locale __cloc, size_t __refs=0)
- int [_M_compare](#) (const `_CharT` *, const `_CharT` *) const throw ()
- template<>
int [_M_compare](#) (const char *, const char *) const throw()
- template<>
int [_M_compare](#) (const `wchar_t` *, const `wchar_t` *) const throw()
- size_t [_M_transform](#) (`_CharT` *, const `_CharT` *, size_t) const throw ()
- template<>
size_t [_M_transform](#) (char *, const char *, size_t) const throw()
- template<>
size_t [_M_transform](#) (`wchar_t` *, const `wchar_t` *, size_t) const throw()
- int [compare](#) (const `_CharT` * __lo1, const `_CharT` * __hi1, const `_CharT` * __lo2, const `_CharT` * __hi2) const
- long [hash](#) (const `_CharT` * __lo, const `_CharT` * __hi) const
- [string_type transform](#) (const `_CharT` * __lo, const `_CharT` * __hi) const

Static Public Attributes

- static [locale::id](#) id

Protected Member Functions

- virtual [~collate](#) ()
- virtual int [do_compare](#) (const `_CharT` * __lo1, const `_CharT` * __hi1, const `_CharT` * __lo2, const `_CharT` * __hi2) const
- virtual long [do_hash](#) (const `_CharT` * __lo, const `_CharT` * __hi) const
- virtual [string_type do_transform](#) (const `_CharT` * __lo, const `_CharT` * __hi) const

Static Protected Member Functions

- static `_c_locale` [_S_clone_c_locale](#) (`_c_locale` & __cloc) throw ()
- static void [_S_create_c_locale](#) (`_c_locale` & __cloc, const char * __s, `_c_locale` __old=0)
- static void [_S_destroy_c_locale](#) (`_c_locale` & __cloc)
- static `_c_locale` [_S_get_c_locale](#) ()
- static const char * [_S_get_c_name](#) () throw ()
- static `_c_locale` [_S_lc_ctype_c_locale](#) (`_c_locale` __cloc, const char * __s)

Protected Attributes

- `_c_locale` [_M_c_locale_collate](#)

5.660.1 Detailed Description

```
template<typename _CharT>
class std::collate<_CharT>
```

Facet for localized string comparison.

This facet encapsulates the code to compare strings in a localized manner.

The collate template uses protected virtual functions to provide the actual results. The public accessors forward the call to the virtual functions. These virtual functions are hooks for developers to implement the behavior they require from the collate facet.

Definition at line 641 of file locale_classes.h.

5.660.2 Member Typedef Documentation

5.660.2.1 template<typename _CharT> typedef _CharT std::collate<_CharT>::char_type

Public typedefs.

Definition at line 647 of file locale_classes.h.

5.660.2.2 template<typename _CharT> typedef basic_string<_CharT> std::collate<_CharT>::string_type

Public typedefs.

Definition at line 648 of file locale_classes.h.

5.660.3 Constructor & Destructor Documentation

5.660.3.1 template<typename _CharT> std::collate<_CharT>::collate (size_t __refs = 0) [inline], [explicit]

Constructor performs initialization.

This is the constructor provided by the standard.

Parameters

<code>__refs</code>	Passed to the base facet class.
---------------------	---------------------------------

Definition at line 668 of file locale_classes.h.

5.660.3.2 template<typename _CharT> std::collate<_CharT>::collate (__c_locale __cloc, size_t __refs = 0) [inline], [explicit]

Internal constructor. Not for general use.

This is a constructor for use by the library itself to set up new locales.

Parameters

<code>__cloc</code>	The C locale.
<code>__refs</code>	Passed to the base facet class.

Definition at line 682 of file `locale_classes.h`.

5.660.3.3 `template<typename _CharT> virtual std::collate<_CharT>::~collate () [inline], [protected], [virtual]`

Destructor.

Definition at line 745 of file `locale_classes.h`.

5.660.4 Member Function Documentation

5.660.4.1 `template<typename _CharT> int std::collate<_CharT>::compare (const _CharT * __lo1, const _CharT * __hi1, const _CharT * __lo2, const _CharT * __hi2) const [inline]`

Compare two strings.

This function compares two strings and returns the result by calling `collate::do_compare()`.

Parameters

<code>__lo1</code>	Start of string 1.
<code>__hi1</code>	End of string 1.
<code>__lo2</code>	Start of string 2.
<code>__hi2</code>	End of string 2.

Returns

1 if `string1 > string2`, -1 if `string1 < string2`, else 0.

Definition at line 699 of file `locale_classes.h`.

5.660.4.2 `template<typename _CharT> int std::collate<_CharT>::do_compare (const _CharT * __lo1, const _CharT * __hi1, const _CharT * __lo2, const _CharT * __hi2) const [protected], [virtual]`

Compare two strings.

This function is a hook for derived classes to change the value returned.

See also

`compare()`.

Parameters

<code>__lo1</code>	Start of string 1.
<code>__hi1</code>	End of string 1.
<code>__lo2</code>	Start of string 2.
<code>__hi2</code>	End of string 2.

Returns

1 if string1 > string2, -1 if string1 < string2, else 0.

Definition at line 161 of file locale_classes.tcc.

References `std::basic_string<_CharT, _Traits, _Alloc>::c_str()`, `std::basic_string<_CharT, _Traits, _Alloc>::data()`, `std::collate<_CharT>::do_transform()`, and `std::basic_string<_CharT, _Traits, _Alloc>::length()`.

Referenced by `std::use_facet()`.

5.660.4.3 `template<typename _CharT> long std::collate<_CharT>::do_hash (const _CharT * __lo, const _CharT * __hi)`
`const [protected], [virtual]`

Return hash of a string.

This function computes and returns a hash on the input string. This function is a hook for derived classes to change the value returned.

Parameters

<code>__↔ __lo</code>	Start of string.
<code>__↔ __hi</code>	End of string.

Returns

Hash value.

Definition at line 256 of file locale_classes.tcc.

Referenced by `std::collate<_CharT>::do_transform()`.

5.660.4.4 `template<typename _CharT> collate<_CharT>::string_type std::collate<_CharT>::do_transform (const _CharT * __lo, const _CharT * __hi) const`
`[protected], [virtual]`

Transform string to comparable form.

This function is a hook for derived classes to change the value returned.

Parameters

\leftrightarrow _lo	Start.
\leftrightarrow _hi	End.

Returns

transformed string.

Definition at line 200 of file locale_classes.tcc.

References `std::basic_string<_CharT, _Traits, _Alloc>::append()`, `std::basic_string<_CharT, _Traits, _Alloc>::c_str()`, `std::basic_string<_CharT, _Traits, _Alloc>::data()`, `std::collate<_CharT>::do_hash()`, `std::basic_string<_CharT, _Traits, _Alloc>::length()`, and `std::basic_string<_CharT, _Traits, _Alloc>::push_back()`.

Referenced by `std::collate<_CharT>::do_compare()`.

5.660.4.5 `template<typename _CharT> long std::collate<_CharT>::hash (const _CharT * __lo, const _CharT * __hi) const [inline]`

Return hash of a string.

This function computes and returns a hash on the input string. It does so by returning `collate::do_hash()`.

Parameters

\leftrightarrow _lo	Start of string.
\leftrightarrow _hi	End of string.

Returns

Hash value.

Definition at line 732 of file locale_classes.h.

5.660.4.6 `template<typename _CharT> string_type std::collate<_CharT>::transform (const _CharT * __lo, const _CharT * __hi) const [inline]`

Transform string to comparable form.

This function is a wrapper for `strxfrm` functionality. It takes the input string and returns a modified string that can be directly compared to other transformed strings. In the C locale, this function just returns a copy of the input string. In some other locales, it may replace two chars with one, change a char for another, etc. It does so by returning `collate::do_transform()`.

Parameters

<code>_↔ _lo</code>	Start of string.
<code>_↔ _hi</code>	End of string.

Returns

Transformed string_type.

Definition at line 718 of file `locale_classes.h`.

5.660.5 Member Data Documentation

5.660.5.1 `template<typename _CharT> locale::id std::collate<_CharT>::id` `[static]`

Numpunct facet id.

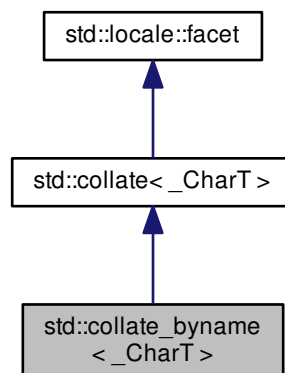
Definition at line 658 of file `locale_classes.h`.

The documentation for this class was generated from the following files:

- [locale_classes.h](#)
- [locale_classes.tcc](#)

5.661 `std::collate_byname<_CharT>` Class Template Reference

Inheritance diagram for `std::collate_byname<_CharT>`:



Public Types

- typedef `_CharT` `char_type`
- typedef `basic_string<_CharT>` `string_type`

Public Member Functions

- `collate_byname` (const char *__s, size_t __refs=0)
- `collate_byname` (const `string` &__s, size_t __refs=0)
- `int _M_compare` (const `_CharT` *, const `_CharT` *) const throw ()
- template<>
 `int _M_compare` (const char *, const char *) const throw()
- template<>
 `int _M_compare` (const `wchar_t` *, const `wchar_t` *) const throw()
- `size_t _M_transform` (`_CharT` *, const `_CharT` *, size_t) const throw ()
- template<>
 `size_t _M_transform` (char *, const char *, size_t) const throw()
- template<>
 `size_t _M_transform` (`wchar_t` *, const `wchar_t` *, size_t) const throw()
- `int compare` (const `_CharT` *__lo1, const `_CharT` *__hi1, const `_CharT` *__lo2, const `_CharT` *__hi2) const
- `long hash` (const `_CharT` *__lo, const `_CharT` *__hi) const
- `string_type transform` (const `_CharT` *__lo, const `_CharT` *__hi) const

Static Public Attributes

- static `locale::id` `id`

Protected Member Functions

- virtual `int do_compare` (const `_CharT` *__lo1, const `_CharT` *__hi1, const `_CharT` *__lo2, const `_CharT` *__hi2) const
- virtual `long do_hash` (const `_CharT` *__lo, const `_CharT` *__hi) const
- virtual `string_type do_transform` (const `_CharT` *__lo, const `_CharT` *__hi) const

Static Protected Member Functions

- static `__c_locale _S_clone_c_locale` (`__c_locale` &__cloc) throw ()
- static void `_S_create_c_locale` (`__c_locale` &__cloc, const char *__s, `__c_locale` __old=0)
- static void `_S_destroy_c_locale` (`__c_locale` &__cloc)
- static `__c_locale _S_get_c_locale` ()
- static const char * `_S_get_c_name` () throw ()
- static `__c_locale _S_lc_ctype_c_locale` (`__c_locale` __cloc, const char *__s)

Protected Attributes

- `__c_locale _M_c_locale_collate`

5.661.1 Detailed Description

```
template<typename _CharT>
class std::collate_byname<_CharT>
```

class collate_byname [22.2.4.2].

Definition at line 815 of file locale_classes.h.

5.661.2 Member Typedef Documentation

5.661.2.1 `template<typename _CharT> typedef _CharT std::collate_byname<_CharT>::char_type`

Public typedefs.

Definition at line 820 of file locale_classes.h.

5.661.2.2 `template<typename _CharT> typedef basic_string<_CharT> std::collate_byname<_CharT>::string_type`

Public typedefs.

Definition at line 821 of file locale_classes.h.

5.661.3 Member Function Documentation

5.661.3.1 `template<typename _CharT> int std::collate<_CharT>::compare (const _CharT * __lo1, const _CharT * __hi1, const _CharT * __lo2, const _CharT * __hi2) const` [inline],[inherited]

Compare two strings.

This function compares two strings and returns the result by calling collate::do_compare().

Parameters

<code>__lo1</code>	Start of string 1.
<code>__hi1</code>	End of string 1.
<code>__lo2</code>	Start of string 2.
<code>__hi2</code>	End of string 2.

Returns

1 if string1 > string2, -1 if string1 < string2, else 0.

Definition at line 699 of file locale_classes.h.

5.661.3.2 `template<typename _CharT> int std::collate<_CharT>::do_compare (const _CharT * __lo1, const _CharT * __hi1, const _CharT * __lo2, const _CharT * __hi2) const` [protected], [virtual], [inherited]

Compare two strings.

This function is a hook for derived classes to change the value returned.

See also

`compare()`.

Parameters

<code>__lo1</code>	Start of string 1.
<code>__hi1</code>	End of string 1.
<code>__lo2</code>	Start of string 2.
<code>__hi2</code>	End of string 2.

Returns

1 if `string1 > string2`, -1 if `string1 < string2`, else 0.

Definition at line 161 of file `locale_classes.tcc`.

References `std::basic_string<_CharT, _Traits, _Alloc>::c_str()`, `std::basic_string<_CharT, _Traits, _Alloc>::data()`, `std::collate<_CharT>::do_transform()`, and `std::basic_string<_CharT, _Traits, _Alloc>::length()`.

Referenced by `std::use_facet()`.

5.661.3.3 `template<typename _CharT> long std::collate<_CharT>::do_hash (const _CharT * __lo, const _CharT * __hi) const` [protected], [virtual], [inherited]

Return hash of a string.

This function computes and returns a hash on the input string. This function is a hook for derived classes to change the value returned.

Parameters

<code>__lo</code>	Start of string.
<code>__hi</code>	End of string.

Returns

Hash value.

Definition at line 256 of file locale_classes.tcc.

Referenced by std::collate<_CharT>::do_transform().

5.661.3.4 `template<typename _CharT> collate<_CharT>::string_type std::collate<_CharT>::do_transform (const _CharT * __lo, const _CharT * __hi) const` `[protected], [virtual], [inherited]`

Transform string to comparable form.

This function is a hook for derived classes to change the value returned.

Parameters

<code>__lo</code>	Start.
<code>__hi</code>	End.

Returns

transformed string.

Definition at line 200 of file locale_classes.tcc.

References std::basic_string<_CharT, _Traits, _Alloc>::append(), std::basic_string<_CharT, _Traits, _Alloc>::c_str(), std::basic_string<_CharT, _Traits, _Alloc>::data(), std::collate<_CharT>::do_hash(), std::basic_string<_CharT, _Traits, _Alloc>::length(), and std::basic_string<_CharT, _Traits, _Alloc>::push_back().

Referenced by std::collate<_CharT>::do_compare().

5.661.3.5 `template<typename _CharT> long std::collate<_CharT>::hash (const _CharT * __lo, const _CharT * __hi) const` `[inline], [inherited]`

Return hash of a string.

This function computes and returns a hash on the input string. It does so by returning collate::do_hash().

Parameters

<code>__lo</code>	Start of string.
<code>__hi</code>	End of string.

Returns

Hash value.

Definition at line 732 of file locale_classes.h.

5.661.3.6 `template<typename _CharT> string_type std::collate<_CharT>::transform (const _CharT * __lo, const _CharT * __hi) const` `[inline],[inherited]`

Transform string to comparable form.

This function is a wrapper for `strxfrm` functionality. It takes the input string and returns a modified string that can be directly compared to other transformed strings. In the C locale, this function just returns a copy of the input string. In some other locales, it may replace two chars with one, change a char for another, etc. It does so by returning `collate::do_transform()`.

Parameters

<code>__lo</code>	Start of string.
<code>__hi</code>	End of string.

Returns

Transformed `string_type`.

Definition at line 718 of file `locale_classes.h`.

5.661.4 Member Data Documentation

5.661.4.1 `template<typename _CharT> locale::id std::collate<_CharT>::id` `[static],[inherited]`

Numpunct facet id.

Definition at line 658 of file `locale_classes.h`.

The documentation for this class was generated from the following file:

- [locale_classes.h](#)

5.662 `std::complex<_Tp>` Struct Template Reference

Public Types

- typedef `_Tp` [value_type](#)

Public Member Functions

- constexpr **complex** (const _Tp &__r=_Tp(), const _Tp &__i=_Tp())
- constexpr **complex** (const **complex** &)=default
- template<typename _Up >
constexpr **complex** (const **complex**<_Up> &__z)
- constexpr **complex** __rep () const
- _GLIBCXX_ABI_TAG_CXX11 constexpr _Tp **imag** () const
- void **imag** (_Tp __val)
- **complex**<_Tp> & **operator*=** (const _Tp &)
- template<typename _Up >
complex<_Tp> & **operator*=** (const **complex**<_Up> &)
- **complex**<_Tp> & **operator+=** (const _Tp &__t)
- template<typename _Up >
complex<_Tp> & **operator+=** (const **complex**<_Up> &)
- **complex**<_Tp> & **operator-=** (const _Tp &__t)
- template<typename _Up >
complex<_Tp> & **operator-=** (const **complex**<_Up> &)
- **complex**<_Tp> & **operator/=** (const _Tp &)
- template<typename _Up >
complex<_Tp> & **operator/=** (const **complex**<_Up> &)
- **complex**<_Tp> & **operator=** (const _Tp &)
- **complex** & **operator=** (const **complex** &)=default
- template<typename _Up >
complex<_Tp> & **operator=** (const **complex**<_Up> &)
- _GLIBCXX_ABI_TAG_CXX11 constexpr _Tp **real** () const
- void **real** (_Tp __val)

5.662.1 Detailed Description

```
template<typename _Tp>
struct std::complex<_Tp>
```

Template to represent complex numbers.

Specializations for float, double, and long double are part of the library. Results with any other type are not guaranteed.

Parameters

<i>Tp</i>	Type of real and imaginary values.
-----------	------------------------------------

Definition at line 63 of file complex.

5.662.2 Member Typedef Documentation

5.662.2.1 template<typename _Tp> typedef _Tp std::complex<_Tp>::value_type

Value typedef.

Definition at line 125 of file complex.

5.662.3 Constructor & Destructor Documentation

5.662.3.1 `template<typename _Tp> constexpr std::complex<_Tp>::complex(const _Tp & __r = _Tp(), const _Tp & __i = _Tp()) [inline]`

Default constructor. First parameter is x, second parameter is y. Unspecified parameters default to 0.

Definition at line 129 of file complex.

5.662.3.2 `template<typename _Tp> template<typename _Up> constexpr std::complex<_Tp>::complex(const complex<_Up> & __z) [inline]`

Converting constructor.

Definition at line 139 of file complex.

5.662.4 Member Function Documentation

5.662.4.1 `template<typename _Tp> complex<_Tp>& std::complex<_Tp>::operator+=(const _Tp & __t) [inline]`

Add a scalar to this complex number.

Definition at line 184 of file complex.

5.662.4.2 `template<typename _Tp> complex<_Tp>& std::complex<_Tp>::operator-=(const _Tp & __t) [inline]`

Subtract a scalar from this complex number.

Definition at line 193 of file complex.

The documentation for this struct was generated from the following file:

- [complex](#)

5.663 std::complex< double > Struct Template Reference

Public Types

- typedef __complex__ double **_ComplexT**
- typedef double **value_type**

Public Member Functions

- constexpr **complex** (_ComplexT __z)
- constexpr **complex** (double __r=0.0, double __i=0.0)
- constexpr **complex** (const **complex**< float > &__z)
- constexpr **complex** (const **complex**< long double > &)
- **__attribute** ((__abi_tag__("cxx11"))) const expr double real() const
- **__attribute** ((__abi_tag__("cxx11"))) const expr double imag() const
- constexpr _ComplexT **__rep** () const
- void **imag** (double __val)
- **complex** & **operator***= (double __d)
- template<typename _Tp >
 complex & **operator***= (const **complex**< _Tp > &__z)
- **complex** & **operator**+= (double __d)
- template<typename _Tp >
 complex & **operator**+= (const **complex**< _Tp > &__z)
- **complex** & **operator**-= (double __d)
- template<typename _Tp >
 complex & **operator**-= (const **complex**< _Tp > &__z)
- **complex** & **operator**/= (double __d)
- template<typename _Tp >
 complex & **operator**/= (const **complex**< _Tp > &__z)
- **complex** & **operator**= (double __d)
- template<typename _Tp >
 complex & **operator**= (const **complex**< _Tp > &__z)
- void **real** (double __val)

5.663.1 Detailed Description

```
template<>
struct std::complex< double >
```

26.2.3 complex specializations complex<double> specialization

Definition at line 1200 of file complex.

The documentation for this struct was generated from the following file:

- [complex](#)

5.664 std::complex< float > Struct Template Reference

Public Types

- typedef __complex__ float **_ComplexT**
- typedef float **value_type**

Public Member Functions

- constexpr **complex** (**_ComplexT** __z)
- constexpr **complex** (float __r=0.0f, float __i=0.0f)
- constexpr **complex** (const **complex**< double > &)
- constexpr **complex** (const **complex**< long double > &)
- **__attribute** ((__abi_tag__("cxx11"))) const expr float real() const
- **__attribute** ((__abi_tag__("cxx11"))) const expr float imag() const
- constexpr **_ComplexT** **__rep** () const
- void **imag** (float __val)
- **complex** & **operator***= (float __f)
- template<class **_Tp** >
 complex & **operator***= (const **complex**< **_Tp** > &__z)
- **complex** & **operator**+= (float __f)
- template<typename **_Tp** >
 complex & **operator**+= (const **complex**< **_Tp** > &__z)
- **complex** & **operator**-= (float __f)
- template<class **_Tp** >
 complex & **operator**-= (const **complex**< **_Tp** > &__z)
- **complex** & **operator**/= (float __f)
- template<class **_Tp** >
 complex & **operator**/= (const **complex**< **_Tp** > &__z)
- **complex** & **operator**= (float __f)
- template<typename **_Tp** >
 complex & **operator**= (const **complex**< **_Tp** > &__z)
- void **real** (float __val)

5.664.1 Detailed Description

```
template<>
struct std::complex< float >
```

26.2.3 complex specializations **complex**<float> specialization

Definition at line 1051 of file **complex**.

The documentation for this struct was generated from the following file:

- **complex**

5.665 **std::complex**< long double > Struct Template Reference

Public Types

- typedef **__complex__** long double **_ComplexT**
- typedef long double **value_type**

Public Member Functions

- constexpr **complex** (`_ComplexT __z`)
- constexpr **complex** (`long double __r=0.0L, long double __i=0.0L`)
- constexpr **complex** (`const complex< float > &__z`)
- constexpr **complex** (`const complex< double > &__z`)
- **__attribute** (`((__abi_tag__("cxx11")))`) const expr long double real() const
- **__attribute** (`((__abi_tag__("cxx11")))`) const expr long double imag() const
- constexpr `_ComplexT __rep` () const
- void **imag** (`long double __val`)
- [complex](#) & **operator*=** (`long double __r`)
- template<typename `_Tp` >
[complex](#) & **operator*=** (`const complex< _Tp > &__z`)
- [complex](#) & **operator+=** (`long double __r`)
- template<typename `_Tp` >
[complex](#) & **operator+=** (`const complex< _Tp > &__z`)
- [complex](#) & **operator-=** (`long double __r`)
- template<typename `_Tp` >
[complex](#) & **operator-=** (`const complex< _Tp > &__z`)
- [complex](#) & **operator/=** (`long double __r`)
- template<typename `_Tp` >
[complex](#) & **operator/=** (`const complex< _Tp > &__z`)
- [complex](#) & **operator=** (`long double __r`)
- template<typename `_Tp` >
[complex](#) & **operator=** (`const complex< _Tp > &__z`)
- void **real** (`long double __val`)

5.665.1 Detailed Description

```
template<>
struct std::complex< long double >
```

26.2.3 complex specializations `complex<long double>` specialization

Definition at line 1350 of file `complex`.

The documentation for this struct was generated from the following file:

- [complex](#)

5.666 std::condition_variable Class Reference

Public Types

- typedef `__native_type *` **native_handle_type**

Public Member Functions

- **condition_variable** (const [condition_variable](#) &)=delete
- native_handle_type **native_handle** ()
- void **notify_all** () noexcept
- void **notify_one** () noexcept
- [condition_variable](#) & **operator=** (const [condition_variable](#) &)=delete
- void **wait** ([unique_lock](#)< [mutex](#) > &__lock) noexcept
- template<typename _Predicate >
void **wait** ([unique_lock](#)< [mutex](#) > &__lock, _Predicate __p)
- template<typename _Rep , typename _Period >
[cv_status](#) **wait_for** ([unique_lock](#)< [mutex](#) > &__lock, const [chrono::duration](#)< _Rep, _Period > &__rtime)
- template<typename _Rep , typename _Period , typename _Predicate >
bool **wait_for** ([unique_lock](#)< [mutex](#) > &__lock, const [chrono::duration](#)< _Rep, _Period > &__rtime, _Predicate __p)
- template<typename _Duration >
[cv_status](#) **wait_until** ([unique_lock](#)< [mutex](#) > &__lock, const [chrono::time_point](#)< __clock_t, _Duration > &__atime)
- template<typename _Clock , typename _Duration >
[cv_status](#) **wait_until** ([unique_lock](#)< [mutex](#) > &__lock, const [chrono::time_point](#)< _Clock, _Duration > &__atime)
- template<typename _Clock , typename _Duration , typename _Predicate >
bool **wait_until** ([unique_lock](#)< [mutex](#) > &__lock, const [chrono::time_point](#)< _Clock, _Duration > &__atime, _Predicate __p)

5.666.1 Detailed Description

[condition_variable](#)

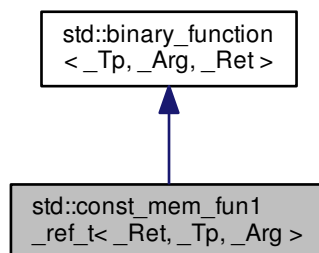
Definition at line 65 of file [condition_variable](#).

The documentation for this class was generated from the following file:

- [condition_variable](#)

5.667 [std::const_mem_fun1_ref_t< _Ret, _Tp, _Arg >](#) Class Template Reference

Inheritance diagram for [std::const_mem_fun1_ref_t< _Ret, _Tp, _Arg >](#):



Public Types

- typedef _Tp [first_argument_type](#)
- typedef _Ret [result_type](#)
- typedef _Arg [second_argument_type](#)

Public Member Functions

- **const_mem_fun1_ref_t** (_Ret(_Tp::*__pf)(_Arg) const)
- **_Ret operator()** (const _Tp &__r, _Arg __x) const

5.667.1 Detailed Description

```
template<typename _Ret, typename _Tp, typename _Arg>
class std::const_mem_fun1_ref_t<_Ret, _Tp, _Arg >
```

One of the [adaptors for member pointers](#).

Definition at line 1064 of file stl_function.h.

5.667.2 Member Typedef Documentation

5.667.2.1 typedef _Tp std::binary_function<_Tp, _Arg, _Ret>::first_argument_type [inherited]

first_argument_type is the type of the first argument

Definition at line 121 of file stl_function.h.

5.667.2.2 typedef _Ret std::binary_function<_Tp, _Arg, _Ret>::result_type [inherited]

result_type is the return type

Definition at line 127 of file stl_function.h.

5.667.2.3 typedef _Arg std::binary_function<_Tp, _Arg, _Ret>::second_argument_type [inherited]

second_argument_type is the type of the second argument

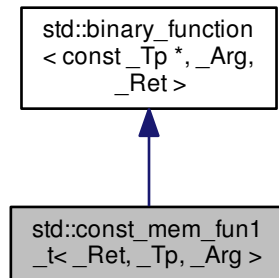
Definition at line 124 of file stl_function.h.

The documentation for this class was generated from the following file:

- [stl_function.h](#)

5.668 std::const_mem_fun1_t< _Ret, _Tp, _Arg > Class Template Reference

Inheritance diagram for std::const_mem_fun1_t< _Ret, _Tp, _Arg >:



Public Types

- typedef const _Tp * [first_argument_type](#)
- typedef _Ret [result_type](#)
- typedef _Arg [second_argument_type](#)

Public Member Functions

- **const_mem_fun1_t** (_Ret(_Tp::*__pf)(_Arg) const)
- _Ret **operator()** (const _Tp *__p, _Arg __x) const

5.668.1 Detailed Description

```
template<typename _Ret, typename _Tp, typename _Arg>
class std::const_mem_fun1_t< _Ret, _Tp, _Arg >
```

One of the [adaptors for member pointers](#).

Definition at line 1028 of file `stl_function.h`.

5.668.2 Member Typedef Documentation

5.668.2.1 typedef const _Tp * **std::binary_function< const_Tp *, _Arg, _Ret >::first_argument_type** [inherited]

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.668.2.2 `typedef _Ret std::binary_function< const_Tp*, _Arg, _Ret>::result_type` [inherited]

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.668.2.3 `typedef _Arg std::binary_function< const_Tp*, _Arg, _Ret>::second_argument_type` [inherited]

`second_argument_type` is the type of the second argument

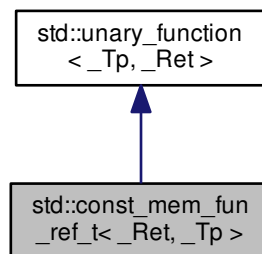
Definition at line 124 of file `stl_function.h`.

The documentation for this class was generated from the following file:

- [stl_function.h](#)

5.669 `std::const_mem_fun_ref_t<_Ret, _Tp>` Class Template Reference

Inheritance diagram for `std::const_mem_fun_ref_t<_Ret, _Tp>`:



Public Types

- `typedef _Tp` [argument_type](#)
- `typedef _Ret` [result_type](#)

Public Member Functions

- `const_mem_fun_ref_t(_Ret(_Tp::*__pf)()) const`
- `_Ret operator() (const _Tp &__r) const`

5.669.1 Detailed Description

```
template<typename _Ret, typename _Tp>
class std::const_mem_fun_ref_t< _Ret, _Tp >
```

One of the [adaptors for member pointers](#).

Definition at line 992 of file `stl_function.h`.

5.669.2 Member Typedef Documentation

5.669.2.1 `typedef _Tp std::unary_function< _Tp, _Ret >::argument_type` [inherited]

`argument_type` is the type of the argument

Definition at line 108 of file `stl_function.h`.

5.669.2.2 `typedef _Ret std::unary_function< _Tp, _Ret >::result_type` [inherited]

`result_type` is the return type

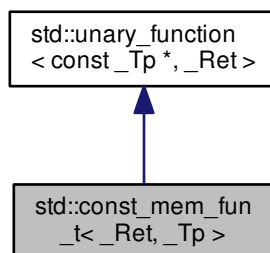
Definition at line 111 of file `stl_function.h`.

The documentation for this class was generated from the following file:

- [stl_function.h](#)

5.670 `std::const_mem_fun_t< _Ret, _Tp >` Class Template Reference

Inheritance diagram for `std::const_mem_fun_t< _Ret, _Tp >`:



Public Types

- typedef `const _Tp *` [argument_type](#)
- typedef `_Ret` [result_type](#)

Public Member Functions

- **`const_mem_fun_t`** (`_Ret(_Tp::*__pf)()` `const`)
- `_Ret` **`operator()`** (`const _Tp *` `__p`) `const`

5.670.1 Detailed Description

```
template<typename _Ret, typename _Tp>  
class std::const_mem_fun_t<_Ret,_Tp>
```

One of the [adaptors for member pointers](#).

Definition at line 956 of file `stl_function.h`.

5.670.2 Member Typedef Documentation

5.670.2.1 `typedef const _Tp *` **`std::unary_function< const _Tp *, _Ret >::argument_type`** `[inherited]`

`argument_type` is the type of the argument

Definition at line 108 of file `stl_function.h`.

5.670.2.2 `typedef _Ret` **`std::unary_function< const _Tp *, _Ret >::result_type`** `[inherited]`

`result_type` is the return type

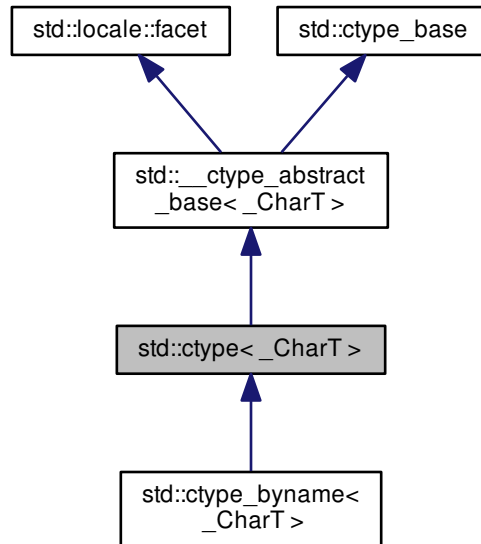
Definition at line 111 of file `stl_function.h`.

The documentation for this class was generated from the following file:

- [stl_function.h](#)

5.671 std::ctype<_CharT> Class Template Reference

Inheritance diagram for std::ctype<_CharT>:



Public Types

- typedef const int * **__to_type**
- typedef _CharT **char_type**
- typedef `__ctype_abstract_base<_CharT>::mask` **mask**

Public Member Functions

- **ctype** (size_t __refs=0)
- bool **is** (mask __m, char_type __c) const
- const char_type * **is** (const char_type * __lo, const char_type * __hi, mask * __vec) const
- char **narrow** (char_type __c, char __default) const
- const char_type * **narrow** (const char_type * __lo, const char_type * __hi, char __default, char * __to) const
- const char_type * **scan_is** (mask __m, const char_type * __lo, const char_type * __hi) const
- const char_type * **scan_not** (mask __m, const char_type * __lo, const char_type * __hi) const
- char_type **tolower** (char_type __c) const
- const char_type * **tolower** (char_type * __lo, const char_type * __hi) const
- char_type **toupper** (char_type __c) const
- const char_type * **toupper** (char_type * __lo, const char_type * __hi) const
- char_type **widen** (char __c) const
- const char * **widen** (const char * __lo, const char * __hi, char_type * __to) const

Static Public Attributes

- static const mask **alnum**
- static const mask **alpha**
- static const mask **blank**
- static const mask **cntrl**
- static const mask **digit**
- static const mask **graph**
- static [locale::id](#) **id**
- static const mask **lower**
- static const mask **print**
- static const mask **punct**
- static const mask **space**
- static const mask **upper**
- static const mask **xdigit**

Protected Member Functions

- virtual bool [do_is](#) (mask __m, [char_type](#) __c) const
- virtual const [char_type](#) * [do_is](#) (const [char_type](#) * __lo, const [char_type](#) * __hi, mask * __vec) const
- virtual [char](#) [do_narrow](#) ([char_type](#), [char](#) __dfault) const
- virtual const [char_type](#) * [do_narrow](#) (const [char_type](#) * __lo, const [char_type](#) * __hi, [char](#) __dfault, [char](#) * __to) const
- virtual const [char_type](#) * [do_scan_is](#) (mask __m, const [char_type](#) * __lo, const [char_type](#) * __hi) const
- virtual const [char_type](#) * [do_scan_not](#) (mask __m, const [char_type](#) * __lo, const [char_type](#) * __hi) const
- virtual [char_type](#) [do_tolower](#) ([char_type](#) __c) const
- virtual const [char_type](#) * [do_tolower](#) ([char_type](#) * __lo, const [char_type](#) * __hi) const
- virtual [char_type](#) [do_toupper](#) ([char_type](#) __c) const
- virtual const [char_type](#) * [do_toupper](#) ([char_type](#) * __lo, const [char_type](#) * __hi) const
- virtual [char_type](#) [do_widen](#) ([char](#) __c) const
- virtual const [char](#) * [do_widen](#) (const [char](#) * __lo, const [char](#) * __hi, [char_type](#) * __dest) const

Static Protected Member Functions

- static [__c_locale](#) [_S_clone_c_locale](#) ([__c_locale](#) & __cloc) throw ()
- static void [_S_create_c_locale](#) ([__c_locale](#) & __cloc, const [char](#) * __s, [__c_locale](#) __old=0)
- static void [_S_destroy_c_locale](#) ([__c_locale](#) & __cloc)
- static [__c_locale](#) [_S_get_c_locale](#) ()
- static const [char](#) * [_S_get_c_name](#) () throw ()
- static [__c_locale](#) [_S_lc_ctype_c_locale](#) ([__c_locale](#) __cloc, const [char](#) * __s)

5.671.1 Detailed Description

```
template<typename _CharT>
class std::ctype< _CharT >
```

Primary class template ctype facet.

This template class defines classification and conversion functions for character sets. It wraps ctype functionality. Ctype gets used by streams for many I/O operations.

This template provides the protected virtual functions the developer will have to replace in a derived class or specialization to make a working facet. The public functions that access them are defined in `__ctype_abstract_base`, to allow for implementation flexibility. See `ctype<wchar_t>` for an example. The functions are documented in `__ctype_abstract_↵` base.

Note: implementations are provided for all the protected virtual functions, but will likely not be useful.

Definition at line 612 of file `locale_facets.h`.

5.671.2 Member Function Documentation

5.671.2.1 `template<typename _CharT> virtual bool std::ctype< _CharT >::do_is (mask __m, char_type __c) const`
`[protected], [virtual]`

Test `char_type` classification.

This function finds a mask `M` for `c` and compares it to mask `m`.

`do_is()` is a hook for a derived facet to change the behavior of classifying. `do_is()` must always return the same result for the same input.

Parameters

<code>↵ __c</code>	The <code>char_type</code> to find the mask of.
<code>↵ __m</code>	The mask to compare against.

Returns

`(M & __m) != 0.`

Implements `std::__ctype_abstract_base< _CharT >`.

5.671.2.2 `template<typename _CharT> virtual const char_type* std::ctype< _CharT >::do_is (const char_type * __lo, const char_type * __hi, mask * __vec) const` `[protected], [virtual]`

Return a mask array.

This function finds the mask for each `char_type` in the range `[lo,hi)` and successively writes it to `vec`. `vec` must have as many elements as the input.

`do_is()` is a hook for a derived facet to change the behavior of classifying. `do_is()` must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__vec</code>	Pointer to an array of mask storage.

Returns

`__hi`.

Implements [std::__ctype_abstract_base<_CharT>](#).

5.671.2.3 `template<typename _CharT> virtual char std::ctype<_CharT>::do_narrow (char_type __c, char __dfault) const [protected], [virtual]`

Narrow `char_type` to `char`.

This virtual function converts the argument to `char` using the simplest reasonable transformation. If the conversion fails, `dfault` is returned instead.

`do_narrow()` is a hook for a derived facet to change the behavior of narrowing. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The <code>char_type</code> to convert.
<code>__dfault</code>	Char to return if conversion fails.

Returns

The converted `char`.

Implements [std::__ctype_abstract_base<_CharT>](#).

5.671.2.4 `template<typename _CharT> virtual const char_type* std::ctype<_CharT>::do_narrow (const char_type * __lo, const char_type * __hi, char __dfault, char * __to) const [protected], [virtual]`

Narrow `char_type` array to `char`.

This virtual function converts each `char_type` in the range `[__lo,__hi)` to `char` using the simplest reasonable transformation and writes the results to the destination array. For any element in the input that cannot be converted, `__default` is used instead.

`do_narrow()` is a hook for a derived facet to change the behavior of narrowing. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__default</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Implements [std::__ctype_abstract_base<_CharT>](#).

5.671.2.5 `template<typename _CharT> virtual const char_type* std::ctype<_CharT>::do_scan_is (mask __m, const char_type * __lo, const char_type * __hi) const` `[protected], [virtual]`

Find `char_type` matching mask.

This function searches for and returns the first `char_type` `c` in `[__lo,__hi)` for which `is(__m,c)` is true.

`do_scan_is()` is a hook for a derived facet to change the behavior of match searching. `do_is()` must always return the same result for the same input.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to a matching `char_type` if found, else `__hi`.

Implements [std::__ctype_abstract_base<_CharT>](#).

5.671.2.6 `template<typename _CharT> virtual const char_type* std::ctype<_CharT>::do_scan_not (mask __m, const char_type * __lo, const char_type * __hi) const` `[protected]`, `[virtual]`

Find char_type not matching mask.

This function searches for and returns a pointer to the first char_type c of [lo,hi) for which is(m,c) is false.

do_scan_is() is a hook for a derived facet to change the behavior of match searching. do_is() must always return the same result for the same input.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to a non-matching char_type if found, else `__hi`.

Implements [std::__ctype_abstract_base<_CharT>](#).

5.671.2.7 `template<typename _CharT> virtual char_type std::ctype<_CharT>::do_tolower (char_type __c) const` `[protected]`, `[virtual]`

Convert to lowercase.

This virtual function converts the argument to lowercase if possible. If not possible (for example, '2'), returns the argument.

do_tolower() is a hook for a derived facet to change the behavior of lowercasing. do_tolower() must always return the same result for the same input.

Parameters

<code>__c</code>	The char_type to convert.
------------------	---------------------------

Returns

The lowercase char_type if convertible, else `__c`.

Implements [std::__ctype_abstract_base<_CharT>](#).

5.671.2.8 `template<typename _CharT> virtual const char_type* std::ctype<_CharT>::do_tolower (char_type * __lo, const char_type * __hi) const` [protected], [virtual]

Convert array to lowercase.

This virtual function converts each `char_type` in the range `[__lo,__hi)` to lowercase if possible. Other elements remain untouched.

`do_tolower()` is a hook for a derived facet to change the behavior of lowercasing. `do_tolower()` must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

Implements [std::__ctype_abstract_base<_CharT>](#).

5.671.2.9 `template<typename _CharT> virtual char_type std::ctype<_CharT>::do_toupper (char_type __c) const` [protected], [virtual]

Convert to uppercase.

This virtual function converts the `char_type` argument to uppercase if possible. If not possible (for example, '2'), returns the argument.

`do_toupper()` is a hook for a derived facet to change the behavior of uppercasing. `do_toupper()` must always return the same result for the same input.

Parameters

<code>__c</code>	The <code>char_type</code> to convert.
------------------	--

Returns

The uppercase `char_type` if convertible, else `__c`.

Implements [std::__ctype_abstract_base<_CharT>](#).

5.671.2.10 `template<typename _CharT> virtual const char_type* std::ctype<_CharT>::do_toupper (char_type * __lo, const char_type * __hi) const` [protected], [virtual]

Convert array to uppercase.

This virtual function converts each `char_type` in the range `[__lo,__hi)` to uppercase if possible. Other elements remain untouched.

`do_toupper()` is a hook for a derived facet to change the behavior of uppercasing. `do_toupper()` must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

Implements [std::__ctype_abstract_base<_CharT>](#).

5.671.2.11 `template<typename _CharT> virtual char_type std::ctype<_CharT>::do_widen (char __c) const` [protected], [virtual]

Widen char.

This virtual function converts the `char` to `char_type` using the simplest reasonable transformation.

`do_widen()` is a hook for a derived facet to change the behavior of widening. `do_widen()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The char to convert.
------------------	----------------------

Returns

The converted `char_type`

Implements [std::__ctype_abstract_base<_CharT>](#).

5.671.2.12 `template<typename _CharT> virtual const char* std::ctype<_CharT>::do_widen (const char * __lo, const char * __hi, char_type * __to) const` `[protected],[virtual]`

Widen char array.

This function converts each char in the input to char_type using the simplest reasonable transformation.

do_widen() is a hook for a derived facet to change the behavior of widening. do_widen() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__lo</code>	Pointer to start range.
<code>__hi</code>	Pointer to end of range.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Implements [std::__ctype_abstract_base<_CharT>](#).

5.671.2.13 `template<typename _CharT> bool std::__ctype_abstract_base<_CharT>::is (mask __m, char_type __c) const` `[inline],[inherited]`

Test char_type classification.

This function finds a mask M for __c and compares it to mask __m. It does so by returning the value of ctype<char_type>::do_is().

Parameters

<code>__c</code>	The char_type to compare the mask of.
<code>__m</code>	The mask to compare against.

Returns

`(M & __m) != 0`.

Definition at line 169 of file locale_facets.h.

Referenced by `std::time_get<_CharT, _InIter>::get()`, `std::ctype<char>::is()`, `std::isalnum()`, `std::isalpha()`, `std::isblank()`, `std::iscntrl()`, `std::isdigit()`, `std::isgraph()`, `std::islower()`, `std::isprint()`, `std::ispunct()`, `std::isspace()`, `std::isupper()`, `std::isxdigit()`, and `std::basic_istream<_CharT, _Traits>::sentry::sentry()`.

5.671.2.14 `template<typename _CharT> const char_type* std::__ctype_abstract_base<_CharT>::is (const char_type
* __lo, const char_type * __hi, mask * __vec) const [inline],[inherited]`

Return a mask array.

This function finds the mask for each char_type in the range [lo,hi) and successively writes it to vec. vec must have as many elements as the char array. It does so by returning the value of ctype<char_type>::do_is().

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__vec</code>	Pointer to an array of mask storage.

Returns

`__hi`.

Definition at line 186 of file locale_facets.h.

5.671.2.15 `template<typename _CharT> char std::__ctype_abstract_base<_CharT>::narrow (char_type __c, char
__dfault) const [inline],[inherited]`

Narrow char_type to char.

This function converts the char_type to char using the simplest reasonable transformation. If the conversion fails, dfault is returned instead. It does so by returning ctype<char_type>::do_narrow(__c).

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__c</code>	The char_type to convert.
<code>__dfault</code>	Char to return if conversion fails.

Returns

The converted char.

Definition at line 331 of file locale_facets.h.

Referenced by std::time_get<_CharT, _InIter>::do_date_order(), std::time_get<_CharT, _InIter>::get(), and std::time_put<_CharT, _OutIter>::put().

5.671.2.16 `template<typename _CharT> const char_type* std::__ctype_abstract_base<_CharT>::narrow (const
char_type * __lo, const char_type * __hi, char __dfault, char * __to) const [inline],[inherited]`

Narrow array to char array.

This function converts each `char_type` in the input to `char` using the simplest reasonable transformation and writes the results to the destination array. For any `char_type` in the input that cannot be converted, *default* is used instead. It does so by returning `ctype<char_type>::do_narrow(__lo, __hi, __default, __to)`.

Note: this is not what you want for codepage conversions. See `codecv` for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__default</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Definition at line 353 of file `locale_facets.h`.

5.671.2.17 `template<typename _CharT> const char_type* std::__ctype_abstract_base<_CharT>::scan_is (mask __m, const char_type * __lo, const char_type * __hi) const` `[inline]`, `[inherited]`

Find `char_type` matching a mask.

This function searches for and returns the first `char_type` `c` in `[lo,hi)` for which `is(m,c)` is true. It does so by returning `ctype<char_type>::do_scan_is()`.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to matching `char_type` if found, else `__hi`.

Definition at line 202 of file `locale_facets.h`.

Referenced by `std::ctype<char>::is()`.

5.671.2.18 `template<typename _CharT> const char_type* std::__ctype_abstract_base<_CharT>::scan_not (mask __m, const char_type * __lo, const char_type * __hi) const` `[inline]`, `[inherited]`

Find `char_type` not matching a mask.

This function searches for and returns the first `char_type` `c` in `[lo,hi)` for which `is(m,c)` is false. It does so by returning `ctype<char_type>::do_scan_not()`.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to first char in range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to non-matching char if found, else `__hi`.

Definition at line 218 of file locale_facets.h.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, and `std::ctype< char >::scan_is()`.

5.671.2.19 `template<typename _CharT> char_type std::__ctype_abstract_base< _CharT >::tolower (char_type __c)`
`const [inline],[inherited]`

Convert to lowercase.

This function converts the argument to lowercase if possible. If not possible (for example, '2'), returns the argument. It does so by returning `ctype<char_type>::do_tolower(c)`.

Parameters

<code>__c</code>	The char_type to convert.
------------------	---------------------------

Returns

The lowercase char_type if convertible, else `__c`.

Definition at line 261 of file locale_facets.h.

Referenced by `std::time_get< _CharT, _InIter >::get()`, and `std::tolower()`.

5.671.2.20 `template<typename _CharT> const char_type* std::__ctype_abstract_base< _CharT >::tolower (char_type`
`* __lo, const char_type * __hi) const [inline],[inherited]`

Convert array to lowercase.

This function converts each char_type in the range `[__lo,__hi)` to lowercase if possible. Other elements remain untouched. It does so by returning `ctype<char_type>::do_tolower(__lo, __hi)`.

Parameters

\leftrightarrow _lo	Pointer to start of range.
\leftrightarrow _hi	Pointer to end of range.

Returns

__hi.

Definition at line 276 of file locale_facets.h.

```
5.671.2.21  template<typename _CharT> char_type std::__ctype_abstract_base<_CharT>::toupper ( char_type __c )
            const [inline],[inherited]
```

Convert to uppercase.

This function converts the argument to uppercase if possible. If not possible (for example, '2'), returns the argument. It does so by returning ctype<char_type>::do_toupper().

Parameters

\leftrightarrow _c	The char_type to convert.
-------------------------	---------------------------

Returns

The uppercase char_type if convertible, else __c.

Definition at line 232 of file locale_facets.h.

Referenced by std::time_get<_CharT, _InIter>::do_date_order(), std::time_get<_CharT, _InIter>::get(), and std::time_get<_CharT, _InIter>::toupper().

```
5.671.2.22  template<typename _CharT> const char_type* std::__ctype_abstract_base<_CharT>::toupper ( char_type
            *__lo, const char_type *__hi ) const [inline],[inherited]
```

Convert array to uppercase.

This function converts each char_type in the range [lo,hi) to uppercase if possible. Other elements remain untouched. It does so by returning ctype<char_type>::do_toupper(lo, hi).

Parameters

\leftrightarrow _lo	Pointer to start of range.
\leftrightarrow _hi	Pointer to end of range.

Returns`__hi`.

Definition at line 247 of file locale_facets.h.

5.671.2.23 `template<typename _CharT> char_type std::__ctype_abstract_base<_CharT>::widen (char __c) const`
`[inline], [inherited]`

Widen char to char_type.

This function converts the char argument to char_type using the simplest reasonable transformation. It does so by returning `ctype<char_type>::do_widen(c)`.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The char to convert.
------------------	----------------------

Returns

The converted char_type.

Definition at line 293 of file locale_facets.h.

Referenced by `std::time_get<_CharT, _InIter>::do_get()`, `std::money_get<_CharT, _InIter>::do_get()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::time_put<_CharT, _OutIter>::do_put()`, `std::money_put<_CharT, _OutIter>::do_put()`, and `std::num_put<_CharT, _OutIter>::do_put()`.

5.671.2.24 `template<typename _CharT> const char* std::__ctype_abstract_base<_CharT>::widen (const char * __lo, const char * __hi, char_type * __to) const` `[inline], [inherited]`

Widen array to char_type.

This function converts each char in the input to char_type using the simplest reasonable transformation. It does so by returning `ctype<char_type>::do_widen(c)`.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Definition at line 312 of file `locale_facets.h`.

5.671.3 Member Data Documentation

5.671.3.1 `template<typename _CharT> locale::id std::ctype<_CharT>::id` `[static]`

The facet id for `ctype<char_type>`

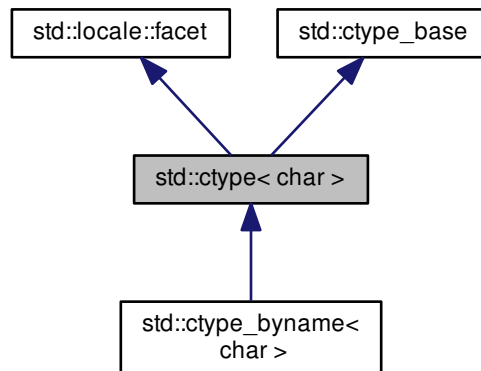
Definition at line 620 of file `locale_facets.h`.

The documentation for this class was generated from the following file:

- [locale_facets.h](#)

5.672 `std::ctype< char >` Class Template Reference

Inheritance diagram for `std::ctype< char >`:



Public Types

- `typedef const int * __to_type`
- `typedef char char_type`
- `typedef unsigned short mask`

Public Member Functions

- `ctype` (const mask * __table=0, bool __del=false, size_t __refs=0)
- `ctype` (__c_locale __cloc, const mask * __table=0, bool __del=false, size_t __refs=0)
- bool `is` (mask __m, char __c) const
- const char * `is` (const char * __lo, const char * __hi, mask * __vec) const
- char `narrow` (char_type __c, char __dfault) const
- const char_type * `narrow` (const char_type * __lo, const char_type * __hi, char __dfault, char * __to) const
- const char * `scan_is` (mask __m, const char * __lo, const char * __hi) const
- const char * `scan_not` (mask __m, const char * __lo, const char * __hi) const
- const mask * `table` () const throw ()
- char_type `tolower` (char_type __c) const
- const char_type * `tolower` (char_type * __lo, const char_type * __hi) const
- char_type `toupper` (char_type __c) const
- const char_type * `toupper` (char_type * __lo, const char_type * __hi) const
- char_type `widen` (char __c) const
- const char * `widen` (const char * __lo, const char * __hi, char_type * __to) const

Static Public Member Functions

- static const mask * `classic_table` () throw ()

Static Public Attributes

- static const mask `alnum`
- static const mask `alpha`
- static const mask `blank`
- static const mask `cntrl`
- static const mask `digit`
- static const mask `graph`
- static `locale::id` `id`
- static const mask `lower`
- static const mask `print`
- static const mask `punct`
- static const mask `space`
- static const size_t `table_size`
- static const mask `upper`
- static const mask `xdigit`

Protected Member Functions

- virtual `~ctype` ()
- virtual char `do_narrow` (char_type __c, char __dfault) const
- virtual const char_type * `do_narrow` (const char_type * __lo, const char_type * __hi, char __dfault, char * __to) const
- virtual char_type `do_tolower` (char_type __c) const
- virtual const char_type * `do_tolower` (char_type * __lo, const char_type * __hi) const
- virtual char_type `do_toupper` (char_type __c) const
- virtual const char_type * `do_toupper` (char_type * __lo, const char_type * __hi) const
- virtual char_type `do_widen` (char __c) const
- virtual const char * `do_widen` (const char * __lo, const char * __hi, char_type * __to) const

Static Protected Member Functions

- static `__c_locale _S_clone_c_locale (__c_locale &__cloc) throw ()`
- static void `_S_create_c_locale (__c_locale &__cloc, const char *__s, __c_locale __old=0)`
- static void `_S_destroy_c_locale (__c_locale &__cloc)`
- static `__c_locale _S_get_c_locale ()`
- static const char * `_S_get_c_name () throw ()`
- static `__c_locale _S_lc_ctype_c_locale (__c_locale __cloc, const char *__s)`

Protected Attributes

- `__c_locale _M_c_locale_ctype`
- bool `_M_del`
- char `_M_narrow [1+static_cast< unsigned char >(-1)]`
- char `_M_narrow_ok`
- const mask * `_M_table`
- `__to_type _M_tolower`
- `__to_type _M_toupper`
- char `_M_widen [1+static_cast< unsigned char >(-1)]`
- char `_M_widen_ok`

5.672.1 Detailed Description

```
template<>
class std::ctype< char >
```

The `ctype<char>` specialization.

This class defines classification and conversion functions for the `char` type. It gets used by `char` streams for many I/O operations. The `char` specialization provides a number of optimizations as well.

Definition at line 681 of file `locale_facets.h`.

5.672.2 Member Typedef Documentation

5.672.2.1 `typedef char std::ctype< char >::char_type`

Typedef for the template parameter `char`.

Definition at line 686 of file `locale_facets.h`.

5.672.3 Constructor & Destructor Documentation

5.672.3.1 `std::ctype< char >::ctype (const mask * __table = 0, bool __del = false, size_t __refs = 0) [explicit]`

Constructor performs initialization.

This is the constructor provided by the standard.

Parameters

<code>__table</code>	If non-zero, table is used as the per-char mask. Else <code>classic_table()</code> is used.
<code>__del</code>	If true, passes ownership of table to this facet.
<code>__refs</code>	Passed to the base facet class.

5.672.3.2 `std::ctype< char >::ctype (__c_locale __cloc, const mask* __table = 0, bool __del = false, size_t __refs = 0)`
`[explicit]`

Constructor performs static initialization.

This constructor is used to construct the initial C locale facet.

Parameters

<code>__cloc</code>	Handle to C locale data.
<code>__table</code>	If non-zero, table is used as the per-char mask.
<code>__del</code>	If true, passes ownership of table to this facet.
<code>__refs</code>	Passed to the base facet class.

5.672.3.3 `virtual std::ctype< char >::~~ctype ()` `[protected], [virtual]`

Destructor.

This function deletes `table()` if `del` was true in the constructor.

5.672.4 Member Function Documentation

5.672.4.1 `static const mask* std::ctype< char >::classic_table () throw` `[static]`

Returns a pointer to the C locale mask table.

5.672.4.2 `virtual char std::ctype< char >::do_narrow (char_type __c, char __dfault) const` `[inline], [protected], [virtual]`

Narrow char.

This virtual function converts the `char` to `char` using the simplest reasonable transformation. If the conversion fails, `dfault` is returned instead. For an undervied `ctype<char>` facet, `c` will be returned unchanged.

`do_narrow()` is a hook for a derived facet to change the behavior of narrowing. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The char to convert.
<code>__default</code>	Char to return if conversion fails.

Returns

The converted char.

Definition at line 1131 of file `locale_facets.h`.

5.672.4.3 `virtual const char_type* std::ctype< char >::do_narrow (const char_type * __lo, const char_type * __hi, char __default, char * __to) const` `[inline], [protected], [virtual]`

Narrow char array to char array.

This virtual function converts each char in the range `[lo,hi)` to char using the simplest reasonable transformation and writes the results to the destination array. For any char in the input that cannot be converted, *default* is used instead. For an underived `ctype<char>` facet, the argument will be copied unchanged.

`do_narrow()` is a hook for a derived facet to change the behavior of narrowing. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__default</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Definition at line 1157 of file `locale_facets.h`.

5.672.4.4 `virtual char_type std::ctype< char >::do_tolower (char_type __c) const` `[protected], [virtual]`

Convert to lowercase.

This virtual function converts the char argument to lowercase if possible. If not possible (for example, '2'), returns the argument.

`do_tolower()` is a hook for a derived facet to change the behavior of lowercasing. `do_tolower()` must always return the same result for the same input.

Parameters

\leftrightarrow _c	The char to convert.
-------------------------	----------------------

Returns

The lowercase char if convertible, else __c.

5.672.4.5 virtual const char_type* std::ctype< char >::do_tolower (char_type * __lo, const char_type * __hi) const [protected], [virtual]

Convert array to lowercase.

This virtual function converts each char in the range [lo,hi) to lowercase if possible. Other chars remain untouched.

do_tolower() is a hook for a derived facet to change the behavior of lowercasing. do_tolower() must always return the same result for the same input.

Parameters

\leftrightarrow _lo	Pointer to first char in range.
\leftrightarrow _hi	Pointer to end of range.

Returns

__hi.

5.672.4.6 virtual char_type std::ctype< char >::do_toupper (char_type __c) const [protected], [virtual]

Convert to uppercase.

This virtual function converts the char argument to uppercase if possible. If not possible (for example, '2'), returns the argument.

do_toupper() is a hook for a derived facet to change the behavior of uppercasing. do_toupper() must always return the same result for the same input.

Parameters

\leftrightarrow _c	The char to convert.
-------------------------	----------------------

Returns

The uppercase char if convertible, else `__c`.

5.672.4.7 `virtual const char_type* std::ctype< char >::do_toupper (char_type * __lo, const char_type * __hi) const` `[protected], [virtual]`

Convert array to uppercase.

This virtual function converts each char in the range `[lo,hi)` to uppercase if possible. Other chars remain untouched.

`do_toupper()` is a hook for a derived facet to change the behavior of uppercasing. `do_toupper()` must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

5.672.4.8 `virtual char_type std::ctype< char >::do_widen (char __c) const` `[inline], [protected], [virtual]`

Widen char.

This virtual function converts the char to char using the simplest reasonable transformation. For an underived `ctype<char>` facet, the argument will be returned unchanged.

`do_widen()` is a hook for a derived facet to change the behavior of widening. `do_widen()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The char to convert.
------------------	----------------------

Returns

The converted character.

Definition at line 1082 of file `locale_facets.h`.

5.672.4.9 `virtual const char* std::ctype< char >::do_widen (const char * __lo, const char * __hi, char_type * __to) const`
`[inline], [protected], [virtual]`

Widen char array.

This function converts each char in the range [lo,hi) to char using the simplest reasonable transformation. For an undervied ctype<char> facet, the argument will be copied unchanged.

do_widen() is a hook for a derived facet to change the behavior of widening. do_widen() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Definition at line 1105 of file locale_facets.h.

5.672.4.10 `bool std::ctype< char >::is (mask __m, char __c) const` `[inline]`

Test char classification.

This function compares the mask table[c] to `__m`.

Parameters

<code>__c</code>	The char to compare the mask of.
<code>__m</code>	The mask to compare against.

Returns

True if `__m & table[__c]` is true, false otherwise.

Definition at line 43 of file ctype_inline.h.

References `std::__ctype_abstract_base< _CharT >::is()`.

5.672.4.11 `const char * std::ctype< char >::is (const char * __lo, const char * __hi, mask * __vec) const` `[inline]`

Return a mask array.

This function finds the mask for each char in the range [lo, hi) and successively writes it to vec. vec must have as many elements as the char array.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__vec</code>	Pointer to an array of mask storage.

Returns

`__hi`.

Definition at line 48 of file `ctype_inline.h`.

References `std::__ctype_abstract_base< _CharT >::scan_is()`.

5.672.4.12 `char std::ctype< char >::narrow (char_type __c, char __dfault) const` `[inline]`

Narrow char.

This function converts the char to char using the simplest reasonable transformation. If the conversion fails, dfault is returned instead. For an underived ctype<char> facet, c will be returned unchanged.

This function works as if it returns `ctype<char>::do_narrow(c)`. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The char to convert.
<code>__dfault</code>	Char to return if conversion fails.

Returns

The converted character.

Definition at line 930 of file `locale_facets.h`.

References `std::__ctype_abstract_base< _CharT >::do_narrow()`.

5.672.4.13 `const char_type* std::ctype< char >::narrow (const char_type * __lo, const char_type * __hi, char __default, char * __to) const` `[inline]`

Narrow char array.

This function converts each char in the input to char using the simplest reasonable transformation and writes the results to the destination array. For any char in the input that cannot be converted, *default* is used instead. For an underived ctype<char> facet, the argument will be copied unchanged.

This function works as if it returns ctype<char>::do_narrow(lo, hi, default, to). do_narrow() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__default</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Definition at line 963 of file locale_facets.h.

References std::__ctype_abstract_base< _CharT >::do_narrow().

5.672.4.14 `const char * std::ctype< char >::scan_is (mask __m, const char * __lo, const char * __hi) const` `[inline]`

Find char matching a mask.

This function searches for and returns the first char in [lo,hi) for which is(m,char) is true.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to a matching char if found, else `__hi`.

Definition at line 57 of file ctype_inline.h.

References `std::__ctype_abstract_base<_CharT>::scan_not()`.

5.672.4.15 `const char * std::ctype<char>::scan_not (mask __m, const char * __lo, const char * __hi) const` `[inline]`

Find char not matching a mask.

This function searches for and returns a pointer to the first char in `[__lo,__hi)` for which `is(m,char)` is false.

Parameters

<code>↔ __m</code>	The mask to compare against.
<code>↔ __lo</code>	Pointer to start of range.
<code>↔ __hi</code>	Pointer to end of range.

Returns

Pointer to a non-matching char if found, else `__hi`.

Definition at line 67 of file `ctype_inline.h`.

5.672.4.16 `const mask* std::ctype<char>::table () const throw` `[inline]`

Returns a pointer to the mask table provided to the constructor, or the default from `classic_table()` if none was provided.

Definition at line 981 of file `locale_facets.h`.

References `std::__ctype_abstract_base<_CharT>::do_tolower()`, and `std::__ctype_abstract_base<_CharT>::do_↔
_toupper()`.

5.672.4.17 `char_type std::ctype<char>::tolower (char_type __c) const` `[inline]`

Convert to lowercase.

This function converts the char argument to lowercase if possible. If not possible (for example, '2'), returns the argument.

`tolower()` acts as if it returns `ctype<char>::do_tolower(__c)`. `do_tolower()` must always return the same result for the same input.

Parameters

<code>↔ __c</code>	The char to convert.
------------------------	----------------------

Returns

The lowercase char if convertible, else `__c`.

Definition at line 835 of file locale_facets.h.

References `std::__ctype_abstract_base<_CharT>::do_tolower()`.

5.672.4.18 `const char_type* std::ctype< char >::tolower (char_type * __lo, const char_type * __hi) const`
`[inline]`

Convert array to lowercase.

This function converts each char in the range [lo,hi) to lowercase if possible. Other chars remain untouched.

`tolower()` acts as if it returns `ctype<char>::do_tolower(__lo, __hi)`. `do_tolower()` must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to first char in range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

Definition at line 852 of file locale_facets.h.

References `std::__ctype_abstract_base<_CharT>::do_tolower()`.

5.672.4.19 `char_type std::ctype< char >::toupper (char_type __c) const` `[inline]`

Convert to uppercase.

This function converts the char argument to uppercase if possible. If not possible (for example, '2'), returns the argument.

`toupper()` acts as if it returns `ctype<char>::do_toupper(c)`. `do_toupper()` must always return the same result for the same input.

Parameters

<code>__c</code>	The char to convert.
------------------	----------------------

Returns

The uppercase char if convertible, else `__c`.

Definition at line 802 of file `locale_facets.h`.

References `std::__ctype_abstract_base<_CharT>::do_toupper()`.

5.672.4.20 `const char_type* std::ctype<char>::toupper (char_type * __lo, const char_type * __hi) const`
`[inline]`

Convert array to uppercase.

This function converts each char in the range `[__lo,__hi)` to uppercase if possible. Other chars remain untouched.

`toupper()` acts as if it returns `ctype<char>::do_toupper(__lo, __hi)`. `do_toupper()` must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to first char in range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

Definition at line 819 of file `locale_facets.h`.

References `std::__ctype_abstract_base<_CharT>::do_toupper()`.

5.672.4.21 `char_type std::ctype<char>::widen (char __c) const` `[inline]`

Widen char.

This function converts the `char` to `char_type` using the simplest reasonable transformation. For an underived `ctype<char>` facet, the argument will be returned unchanged.

This function works as if it returns `ctype<char>::do_widen(c)`. `do_widen()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The char to convert.
------------------	----------------------

Returns

The converted character.

Definition at line 872 of file locale_facets.h.

References std::__ctype_abstract_base< _CharT >::do_widen().

5.672.4.22 `const char* std::ctype< char >::widen (const char * __lo, const char * __hi, char_type * __to) const`
`[inline]`

Widen char array.

This function converts each char in the input to char using the simplest reasonable transformation. For an underived ctype<char> facet, the argument will be copied unchanged.

This function works as if it returns ctype<char>::do_widen(c). do_widen() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__lo</code>	Pointer to first char in range.
<code>__hi</code>	Pointer to end of range.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Definition at line 899 of file locale_facets.h.

References std::__ctype_abstract_base< _CharT >::do_widen().

5.672.5 Member Data Documentation

5.672.5.1 `locale::id std::ctype< char >::id` `[static]`

The facet id for ctype<char>

Definition at line 703 of file locale_facets.h.

5.672.5.2 `const size_t std::ctype< char >::table_size` [static]

The size of the mask table. It is `SCHAR_MAX + 1`.

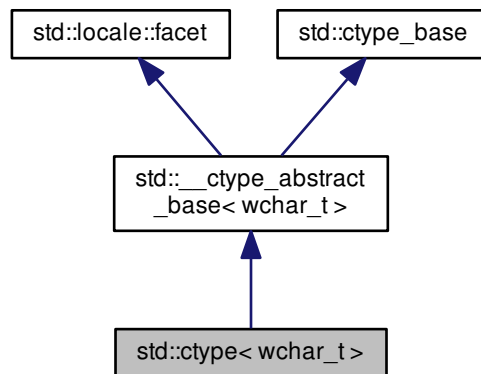
Definition at line 705 of file `locale_facets.h`.

The documentation for this class was generated from the following files:

- [locale_facets.h](#)
- [ctype_inline.h](#)

5.673 `std::ctype< wchar_t >` Class Template Reference

Inheritance diagram for `std::ctype< wchar_t >`:



Public Types

- `typedef const int * __to_type`
- `typedef wctype_t __wmask_type`
- `typedef wchar_t char_type`
- `typedef unsigned short mask`

Public Member Functions

- [ctype](#) (size_t __refs=0)
- [ctype](#) (__c_locale __cloc, size_t __refs=0)
- bool [is](#) (mask __m, [char_type](#) __c) const
- const [char_type](#) * [is](#) (const [char_type](#) * __lo, const [char_type](#) * __hi, mask * __vec) const
- char [narrow](#) ([char_type](#) __c, char __dfault) const
- const [char_type](#) * [narrow](#) (const [char_type](#) * __lo, const [char_type](#) * __hi, char __dfault, char * __to) const
- const [char_type](#) * [scan_is](#) (mask __m, const [char_type](#) * __lo, const [char_type](#) * __hi) const
- const [char_type](#) * [scan_not](#) (mask __m, const [char_type](#) * __lo, const [char_type](#) * __hi) const
- [char_type](#) [tolower](#) ([char_type](#) __c) const
- const [char_type](#) * [tolower](#) ([char_type](#) * __lo, const [char_type](#) * __hi) const
- [char_type](#) [toupper](#) ([char_type](#) __c) const
- const [char_type](#) * [toupper](#) ([char_type](#) * __lo, const [char_type](#) * __hi) const
- [char_type](#) [widen](#) (char __c) const
- const char * [widen](#) (const char * __lo, const char * __hi, [char_type](#) * __to) const

Static Public Attributes

- static const mask **alnum**
- static const mask **alpha**
- static const mask **blank**
- static const mask **cntrl**
- static const mask **digit**
- static const mask **graph**
- static [locale::id](#) **id**
- static const mask **lower**
- static const mask **print**
- static const mask **punct**
- static const mask **space**
- static const mask **upper**
- static const mask **xdigit**

Protected Member Functions

- virtual [~ctype](#) ()
- [__wmask_type](#) **M_convert_to_wmask** (const mask __m) const throw ()
- void **M_initialize_ctype** () throw ()
- virtual bool [do_is](#) (mask __m, [char_type](#) __c) const
- virtual const [char_type](#) * [do_is](#) (const [char_type](#) * __lo, const [char_type](#) * __hi, mask * __vec) const
- virtual char [do_narrow](#) ([char_type](#) __c, char __dfault) const
- virtual const [char_type](#) * [do_narrow](#) (const [char_type](#) * __lo, const [char_type](#) * __hi, char __dfault, char * __to) const
- virtual const [char_type](#) * [do_scan_is](#) (mask __m, const [char_type](#) * __lo, const [char_type](#) * __hi) const
- virtual const [char_type](#) * [do_scan_not](#) (mask __m, const [char_type](#) * __lo, const [char_type](#) * __hi) const
- virtual [char_type](#) [do_tolower](#) ([char_type](#) __c) const
- virtual const [char_type](#) * [do_tolower](#) ([char_type](#) * __lo, const [char_type](#) * __hi) const
- virtual [char_type](#) [do_toupper](#) ([char_type](#) __c) const
- virtual const [char_type](#) * [do_toupper](#) ([char_type](#) * __lo, const [char_type](#) * __hi) const
- virtual [char_type](#) [do_widen](#) (char __c) const
- virtual const char * [do_widen](#) (const char * __lo, const char * __hi, [char_type](#) * __to) const

Static Protected Member Functions

- static `__c_locale _S_clone_c_locale (__c_locale &__cloc) throw ()`
- static void `_S_create_c_locale (__c_locale &__cloc, const char *__s, __c_locale __old=0)`
- static void `_S_destroy_c_locale (__c_locale &__cloc)`
- static `__c_locale _S_get_c_locale ()`
- static const char * `_S_get_c_name () throw ()`
- static `__c_locale _S_lc_ctype_c_locale (__c_locale __cloc, const char *__s)`

Protected Attributes

- mask `_M_bit` [16]
- `__c_locale _M_c_locale_ctype`
- char `_M_narrow` [128]
- bool `_M_narrow_ok`
- `wint_t _M_widen` [1+static_cast< unsigned char >(-1)]
- `__wmask_type _M_wmask` [16]

5.673.1 Detailed Description

```
template<>
class std::ctype< wchar_t >
```

The `ctype<wchar_t>` specialization.

This class defines classification and conversion functions for the `wchar_t` type. It gets used by `wchar_t` streams for many I/O operations. The `wchar_t` specialization provides a number of optimizations as well.

`ctype<wchar_t>` inherits its public methods from `__ctype_abstract_base<wchar_t>`.

Definition at line 1182 of file `locale_facets.h`.

5.673.2 Member Typedef Documentation

5.673.2.1 `typedef wchar_t std::ctype< wchar_t >::char_type`

Typedef for the template parameter `wchar_t`.

Definition at line 1187 of file `locale_facets.h`.

5.673.3 Constructor & Destructor Documentation

5.673.3.1 `std::ctype< wchar_t >::ctype (size_t __refs = 0) [explicit]`

Constructor performs initialization.

This is the constructor provided by the standard.

Parameters

<code>__refs</code>	Passed to the base facet class.
---------------------	---------------------------------

5.673.3.2 `std::ctype< wchar_t >::ctype (__c_locale __cloc, size_t __refs = 0)` `[explicit]`

Constructor performs static initialization.

This constructor is used to construct the initial C locale facet.

Parameters

<code>__cloc</code>	Handle to C locale data.
<code>__refs</code>	Passed to the base facet class.

5.673.3.3 `virtual std::ctype< wchar_t >::~~ctype ()` `[protected]`, `[virtual]`

Destructor.

5.673.4 Member Function Documentation

5.673.4.1 `virtual bool std::ctype< wchar_t >::do_is (mask __m, char_type __c) const` `[protected]`, `[virtual]`

Test `wchar_t` classification.

This function finds a mask `M` for `c` and compares it to mask `m`.

`do_is()` is a hook for a derived facet to change the behavior of classifying. `do_is()` must always return the same result for the same input.

Parameters

<code>__c</code>	The <code>wchar_t</code> to find the mask of.
<code>__m</code>	The mask to compare against.

Returns

`(M & __m) != 0.`

Implements [std::__ctype_abstract_base< wchar_t >](#).

5.673.4.2 `virtual const char_type* std::ctype< wchar_t >::do_is (const char_type * __lo, const char_type * __hi, mask * __vec) const` [protected], [virtual]

Return a mask array.

This function finds the mask for each `wchar_t` in the range `[lo,hi)` and successively writes it to `vec`. `vec` must have as many elements as the input.

`do_is()` is a hook for a derived facet to change the behavior of classifying. `do_is()` must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__vec</code>	Pointer to an array of mask storage.

Returns

`__hi`.

Implements [std::__ctype_abstract_base< wchar_t >](#).

5.673.4.3 `virtual char std::ctype< wchar_t >::do_narrow (char_type __c, char __dfault) const` [protected], [virtual]

Narrow `wchar_t` to `char`.

This virtual function converts the argument to `char` using the simplest reasonable transformation. If the conversion fails, `dfault` is returned instead. For an underived `ctype<wchar_t>` facet, `c` will be cast to `char` and returned.

`do_narrow()` is a hook for a derived facet to change the behavior of narrowing. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The <code>wchar_t</code> to convert.
<code>__dfault</code>	Char to return if conversion fails.

Returns

The converted `char`.

Implements [std::__ctype_abstract_base< wchar_t >](#).

5.673.4.4 `virtual const char_type* std::ctype< wchar_t >::do_narrow (const char_type * __lo, const char_type * __hi, char __dfault, char * __to) const` [protected], [virtual]

Narrow wchar_t array to char array.

This virtual function converts each wchar_t in the range [lo,hi) to char using the simplest reasonable transformation and writes the results to the destination array. For any wchar_t in the input that cannot be converted, *dfault* is used instead. For an underived ctype<wchar_t> facet, the argument will be copied, casting each element to char.

do_narrow() is a hook for a derived facet to change the behavior of narrowing. do_narrow() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__dfault</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Implements [std::__ctype_abstract_base< wchar_t >](#).

5.673.4.5 `virtual const char_type* std::ctype< wchar_t >::do_scan_is (mask __m, const char_type * __lo, const char_type * __hi) const` [protected], [virtual]

Find wchar_t matching mask.

This function searches for and returns the first wchar_t c in [__lo,__hi) for which is(__m,c) is true.

do_scan_is() is a hook for a derived facet to change the behavior of match searching. do_is() must always return the same result for the same input.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to a matching `wchar_t` if found, else `__hi`.

Implements [std::__ctype_abstract_base< wchar_t >](#).

5.673.4.6 `virtual const char_type* std::ctype< wchar_t >::do_scan_not (mask __m, const char_type * __lo, const char_type * __hi) const` `[protected]`, `[virtual]`

Find `wchar_t` not matching mask.

This function searches for and returns a pointer to the first `wchar_t` `c` of `[__lo,__hi)` for which `is(__m,c)` is false.

`do_scan_is()` is a hook for a derived facet to change the behavior of match searching. `do_is()` must always return the same result for the same input.

Parameters

<code>__↔ __m</code>	The mask to compare against.
<code>__↔ __lo</code>	Pointer to start of range.
<code>__↔ __hi</code>	Pointer to end of range.

Returns

Pointer to a non-matching `wchar_t` if found, else `__hi`.

Implements [std::__ctype_abstract_base< wchar_t >](#).

5.673.4.7 `virtual char_type std::ctype< wchar_t >::do_tolower (char_type __c) const` `[protected]`, `[virtual]`

Convert to lowercase.

This virtual function converts the argument to lowercase if possible. If not possible (for example, '2'), returns the argument.

`do_tolower()` is a hook for a derived facet to change the behavior of lowercasing. `do_tolower()` must always return the same result for the same input.

Parameters

<code>__↔ __c</code>	The <code>wchar_t</code> to convert.
--------------------------	--------------------------------------

Returns

The lowercase `wchar_t` if convertible, else `__c`.

Implements [std::__ctype_abstract_base< wchar_t >](#).

5.673.4.8 `virtual const char_type* std::ctype< wchar_t >::do_tolower (char_type * __lo, const char_type * __hi) const` [protected], [virtual]

Convert array to lowercase.

This virtual function converts each `wchar_t` in the range `[lo,hi)` to lowercase if possible. Other elements remain untouched.

`do_tolower()` is a hook for a derived facet to change the behavior of lowercasing. `do_tolower()` must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

Implements [std::__ctype_abstract_base< wchar_t >](#).

5.673.4.9 `virtual char_type std::ctype< wchar_t >::do_toupper (char_type __c) const` [protected], [virtual]

Convert to uppercase.

This virtual function converts the `wchar_t` argument to uppercase if possible. If not possible (for example, '2'), returns the argument.

`do_toupper()` is a hook for a derived facet to change the behavior of uppercasing. `do_toupper()` must always return the same result for the same input.

Parameters

<code>__c</code>	The <code>wchar_t</code> to convert.
------------------	--------------------------------------

Returns

The uppercase `wchar_t` if convertible, else `__c`.

Implements [std::__ctype_abstract_base< wchar_t >](#).

5.673.4.10 `virtual const char_type* std::ctype< wchar_t >::do_toupper (char_type * __lo, const char_type * __hi) const` `[protected], [virtual]`

Convert array to uppercase.

This virtual function converts each `wchar_t` in the range `[lo,hi)` to uppercase if possible. Other elements remain untouched.

`do_toupper()` is a hook for a derived facet to change the behavior of uppercasing. `do_toupper()` must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

Implements [std::__ctype_abstract_base< wchar_t >](#).

5.673.4.11 `virtual char_type std::ctype< wchar_t >::do_widen (char __c) const` `[protected], [virtual]`

Widen char to `wchar_t`.

This virtual function converts the `char` to `wchar_t` using the simplest reasonable transformation. For an underived `ctype<wchar_t>` facet, the argument will be cast to `wchar_t`.

`do_widen()` is a hook for a derived facet to change the behavior of widening. `do_widen()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The char to convert.
------------------	----------------------

Returns

The converted `wchar_t`.

Implements [std::__ctype_abstract_base< wchar_t >](#).

5.673.4.12 `virtual const char* std::ctype< wchar_t >::do_widen (const char * __lo, const char * __hi, char_type * __to) const` `[protected]`, `[virtual]`

Widen char array to wchar_t array.

This function converts each char in the input to wchar_t using the simplest reasonable transformation. For an underived ctype<wchar_t> facet, the argument will be copied, casting each element to wchar_t.

do_widen() is a hook for a derived facet to change the behavior of widening. do_widen() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__lo</code>	Pointer to start range.
<code>__hi</code>	Pointer to end of range.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Implements [std::__ctype_abstract_base< wchar_t >](#).

5.673.4.13 `bool std::__ctype_abstract_base< wchar_t >::is (mask __m, char_type __c) const` `[inline]`, `[inherited]`

Test char_type classification.

This function finds a mask M for __c and compares it to mask __m. It does so by returning the value of ctype<char_type>::do_is().

Parameters

<code>__c</code>	The char_type to compare the mask of.
<code>__m</code>	The mask to compare against.

Returns

`(M & __m) != 0`.

Definition at line 169 of file locale_facets.h.

References [std::__ctype_abstract_base< _CharT >::do_is\(\)](#).

5.673.4.14 `const char_type* std::__ctype_abstract_base<wchar_t>::is(const char_type * __lo, const char_type * __hi, mask * __vec) const` [inline],[inherited]

Return a mask array.

This function finds the mask for each char_type in the range [lo,hi) and successively writes it to vec. vec must have as many elements as the char array. It does so by returning the value of ctype<char_type>::do_is().

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__vec</code>	Pointer to an array of mask storage.

Returns

`__hi`.

Definition at line 186 of file locale_facets.h.

References std::__ctype_abstract_base<_CharT>::do_is().

5.673.4.15 `char std::__ctype_abstract_base<wchar_t>::narrow(char_type __c, char __dfault) const` [inline],[inherited]

Narrow char_type to char.

This function converts the char_type to char using the simplest reasonable transformation. If the conversion fails, dfault is returned instead. It does so by returning ctype<char_type>::do_narrow(__c).

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__c</code>	The char_type to convert.
<code>__dfault</code>	Char to return if conversion fails.

Returns

The converted char.

Definition at line 331 of file locale_facets.h.

References std::__ctype_abstract_base<_CharT>::do_narrow().

5.673.4.16 `const char_type* std::__ctype_abstract_base< wchar_t >::narrow (const char_type * __lo, const char_type * __hi, char __default, char * __to) const` `[inline],[inherited]`

Narrow array to char array.

This function converts each `char_type` in the input to `char` using the simplest reasonable transformation and writes the results to the destination array. For any `char_type` in the input that cannot be converted, *default* is used instead. It does so by returning `ctype<char_type>::do_narrow(__lo, __hi, __default, __to)`.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__default</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Definition at line 353 of file `locale_facets.h`.

References `std::__ctype_abstract_base< _CharT >::do_is()`, `std::__ctype_abstract_base< _CharT >::do_narrow()`, `std::__ctype_abstract_base< _CharT >::do_scan_is()`, `std::__ctype_abstract_base< _CharT >::do_scan_not()`, `std::__ctype_abstract_base< _CharT >::do_tolower()`, `std::__ctype_abstract_base< _CharT >::do_toupper()`, `std::__ctype_abstract_base< _CharT >::do_widen()`, and `std::locale::facet::facet()`.

5.673.4.17 `const char_type* std::__ctype_abstract_base< wchar_t >::scan_is (mask __m, const char_type * __lo, const char_type * __hi) const` `[inline],[inherited]`

Find `char_type` matching a mask.

This function searches for and returns the first `char_type` `c` in `[lo,hi)` for which `is(m,c)` is true. It does so by returning `ctype<char_type>::do_scan_is()`.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to matching char_type if found, else __hi.

Definition at line 202 of file locale_facets.h.

References std::__ctype_abstract_base<_CharT>::do_scan_is().

5.673.4.18 `const char_type* std::__ctype_abstract_base<wchar_t>::scan_not (mask __m, const char_type * __lo, const char_type * __hi) const` [inline],[inherited]

Find char_type not matching a mask.

This function searches for and returns the first char_type c in [lo,hi) for which is(m,c) is false. It does so by returning ctype<char_type>::do_scan_not().

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to first char in range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to non-matching char if found, else __hi.

Definition at line 218 of file locale_facets.h.

References std::__ctype_abstract_base<_CharT>::do_scan_not().

5.673.4.19 `char_type std::__ctype_abstract_base<wchar_t>::tolower (char_type __c) const` [inline],[inherited]

Convert to lowercase.

This function converts the argument to lowercase if possible. If not possible (for example, '2'), returns the argument. It does so by returning ctype<char_type>::do_tolower(c).

Parameters

<code>__c</code>	The char_type to convert.
------------------	---------------------------

Returns

The lowercase char_type if convertible, else __c.

Definition at line 261 of file locale_facets.h.

References std::__ctype_abstract_base<_CharT>::do_tolower().

5.673.4.20 `const char_type* std::__ctype_abstract_base< wchar_t >::tolower (char_type * __lo, const char_type * __hi) const` `[inline]`, `[inherited]`

Convert array to lowercase.

This function converts each char_type in the range [__lo,__hi) to lowercase if possible. Other elements remain untouched. It does so by returning ctype<char_type>::do_tolower(__lo,__hi).

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

Definition at line 276 of file locale_facets.h.

References std::__ctype_abstract_base<_CharT>::do_tolower().

5.673.4.21 `char_type std::__ctype_abstract_base< wchar_t >::toupper (char_type __c) const` `[inline]`, `[inherited]`

Convert to uppercase.

This function converts the argument to uppercase if possible. If not possible (for example, '2'), returns the argument. It does so by returning ctype<char_type>::do_toupper().

Parameters

<code>__c</code>	The char_type to convert.
------------------	---------------------------

Returns

The uppercase char_type if convertible, else __c.

Definition at line 232 of file locale_facets.h.

References `std::__ctype_abstract_base<_CharT>::do_toupper()`.

5.673.4.22 `const char_type* std::__ctype_abstract_base<wchar_t>::toupper (char_type * __lo, const char_type * __hi) const` `[inline], [inherited]`

Convert array to uppercase.

This function converts each `char_type` in the range `[lo,hi)` to uppercase if possible. Other elements remain untouched. It does so by returning `ctype<char_type>::do_toupper(lo, hi)`.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

Definition at line 247 of file `locale_facets.h`.

References `std::__ctype_abstract_base<_CharT>::do_toupper()`.

5.673.4.23 `char_type std::__ctype_abstract_base<wchar_t>::widen (char __c) const` `[inline], [inherited]`

Widen char to `char_type`.

This function converts the `char` argument to `char_type` using the simplest reasonable transformation. It does so by returning `ctype<char_type>::do_widen(c)`.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The char to convert.
------------------	----------------------

Returns

The converted `char_type`.

Definition at line 293 of file `locale_facets.h`.

References `std::__ctype_abstract_base<_CharT>::do_widen()`.

5.673.4.24 `const char* std::__ctype_abstract_base<wchar_t>::widen (const char * __lo, const char * __hi, char_type * __to) const` `[inline],[inherited]`

Widen array to char_type.

This function converts each char in the input to char_type using the simplest reasonable transformation. It does so by returning `ctype<char_type>::do_widen(c)`.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Definition at line 312 of file `locale_facets.h`.

References `std::__ctype_abstract_base<_CharT>::do_widen()`.

5.673.5 Member Data Documentation

5.673.5.1 `locale::id std::ctype<wchar_t>::id` `[static]`

The facet id for `ctype<wchar_t>`

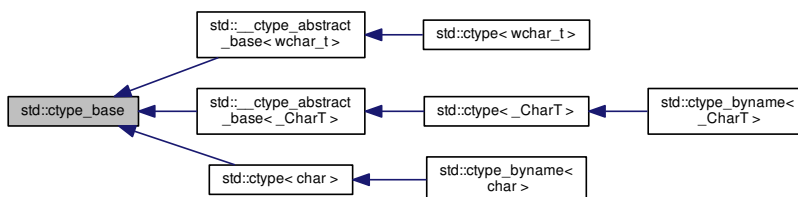
Definition at line 1205 of file `locale_facets.h`.

The documentation for this class was generated from the following file:

- [locale_facets.h](#)

5.674 std::ctype_base Struct Reference

Inheritance diagram for `std::ctype_base`:



Public Types

- typedef const int * **__to_type**
- typedef unsigned short **mask**

Static Public Attributes

- static const mask **alnum**
- static const mask **alpha**
- static const mask **blank**
- static const mask **cntrl**
- static const mask **digit**
- static const mask **graph**
- static const mask **lower**
- static const mask **print**
- static const mask **punct**
- static const mask **space**
- static const mask **upper**
- static const mask **xdigit**

5.674.1 Detailed Description

Base class for ctype.

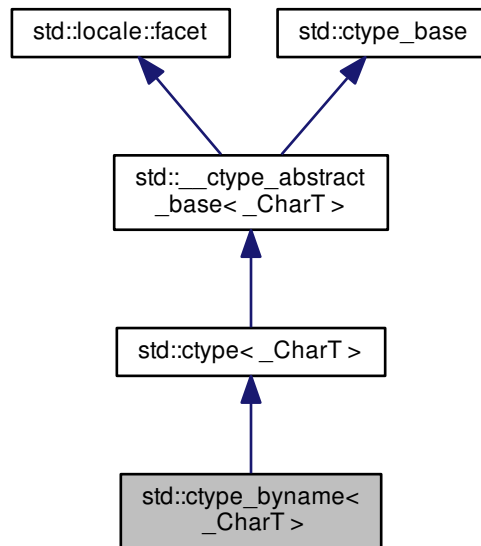
Definition at line 41 of file ctype_base.h.

The documentation for this struct was generated from the following file:

- [ctype_base.h](#)

5.675 std::ctype_byname<_CharT> Class Template Reference

Inheritance diagram for std::ctype_byname<_CharT>:



Public Types

- typedef const int * **__to_type**
- typedef _CharT **char_type**
- typedef `ctype<_CharT>::mask` **mask**

Public Member Functions

- **ctype_byname** (const char * __s, size_t __refs=0)
- **ctype_byname** (const string & __s, size_t __refs=0)
- bool **is** (mask __m, char_type __c) const
- const char_type * **is** (const char_type * __lo, const char_type * __hi, mask * __vec) const
- char **narrow** (char_type __c, char __default) const
- const char_type * **narrow** (const char_type * __lo, const char_type * __hi, char __default, char * __to) const
- const char_type * **scan_is** (mask __m, const char_type * __lo, const char_type * __hi) const
- const char_type * **scan_not** (mask __m, const char_type * __lo, const char_type * __hi) const
- char_type **tolower** (char_type __c) const
- const char_type * **tolower** (char_type * __lo, const char_type * __hi) const
- char_type **toupper** (char_type __c) const
- const char_type * **toupper** (char_type * __lo, const char_type * __hi) const
- char_type **widen** (char __c) const
- const char * **widen** (const char * __lo, const char * __hi, char_type * __to) const

Static Public Attributes

- static const mask **alnum**
- static const mask **alpha**
- static const mask **blank**
- static const mask **cntrl**
- static const mask **digit**
- static const mask **graph**
- static [locale::id](#) **id**
- static const mask **lower**
- static const mask **print**
- static const mask **punct**
- static const mask **space**
- static const mask **upper**
- static const mask **xdigit**

Protected Member Functions

- virtual bool [do_is](#) (mask __m, [char_type](#) __c) const
- virtual const [char_type](#) * [do_is](#) (const [char_type](#) * __lo, const [char_type](#) * __hi, mask * __vec) const
- virtual [char](#) [do_narrow](#) ([char_type](#), [char](#) __dfault) const
- virtual const [char_type](#) * [do_narrow](#) (const [char_type](#) * __lo, const [char_type](#) * __hi, [char](#) __dfault, [char](#) * __to) const
- virtual const [char_type](#) * [do_scan_is](#) (mask __m, const [char_type](#) * __lo, const [char_type](#) * __hi) const
- virtual const [char_type](#) * [do_scan_not](#) (mask __m, const [char_type](#) * __lo, const [char_type](#) * __hi) const
- virtual [char_type](#) [do_tolower](#) ([char_type](#) __c) const
- virtual const [char_type](#) * [do_tolower](#) ([char_type](#) * __lo, const [char_type](#) * __hi) const
- virtual [char_type](#) [do_toupper](#) ([char_type](#) __c) const
- virtual const [char_type](#) * [do_toupper](#) ([char_type](#) * __lo, const [char_type](#) * __hi) const
- virtual [char_type](#) [do_widen](#) ([char](#) __c) const
- virtual const [char](#) * [do_widen](#) (const [char](#) * __lo, const [char](#) * __hi, [char_type](#) * __dest) const

Static Protected Member Functions

- static __c_locale **_S_clone_c_locale** (__c_locale & __cloc) throw ()
- static void **_S_create_c_locale** (__c_locale & __cloc, const [char](#) * __s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale & __cloc)
- static __c_locale **_S_get_c_locale** ()
- static const [char](#) * **_S_get_c_name** () throw ()
- static __c_locale **_S_lc_ctype_c_locale** (__c_locale __cloc, const [char](#) * __s)

5.675.1 Detailed Description

```
template<typename _CharT>
class std::ctype_byname<_CharT>
```

class [ctype_byname](#) [22.2.1.2].

Definition at line 1474 of file [locale_facets.h](#).

5.675.2 Member Function Documentation

5.675.2.1 `template<typename _CharT> virtual bool std::ctype<_CharT>::do_is (mask __m, char_type __c) const`
`[protected], [virtual], [inherited]`

Test char_type classification.

This function finds a mask *M* for *c* and compares it to mask *m*.

do_is() is a hook for a derived facet to change the behavior of classifying. do_is() must always return the same result for the same input.

Parameters

<code>__c</code>	The char_type to find the mask of.
<code>__m</code>	The mask to compare against.

Returns

$(M \& \text{__m}) \neq 0$.

Implements [std::__ctype_abstract_base<_CharT>](#).

5.675.2.2 `template<typename _CharT> virtual const char_type* std::ctype<_CharT>::do_is (const char_type * __lo, const char_type * __hi, mask * __vec) const`
`[protected], [virtual], [inherited]`

Return a mask array.

This function finds the mask for each char_type in the range [lo,hi) and successively writes it to vec. vec must have as many elements as the input.

do_is() is a hook for a derived facet to change the behavior of classifying. do_is() must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__vec</code>	Pointer to an array of mask storage.

Returns

`__hi`.

Implements [std::__ctype_abstract_base<_CharT>](#).

5.675.2.3 `template<typename _CharT> virtual char std::ctype<_CharT>::do_narrow (char_type __c, char __dfault) const`
`[protected], [virtual], [inherited]`

Narrow `char_type` to `char`.

This virtual function converts the argument to `char` using the simplest reasonable transformation. If the conversion fails, `dfault` is returned instead.

`do_narrow()` is a hook for a derived facet to change the behavior of narrowing. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The <code>char_type</code> to convert.
<code>__dfault</code>	Char to return if conversion fails.

Returns

The converted `char`.

Implements [std::__ctype_abstract_base<_CharT>](#).

5.675.2.4 `template<typename _CharT> virtual const char_type* std::ctype<_CharT>::do_narrow (const char_type* __lo, const char_type* __hi, char __dfault, char* __to) const`
`[protected], [virtual], [inherited]`

Narrow `char_type` array to `char`.

This virtual function converts each `char_type` in the range `[__lo, __hi)` to `char` using the simplest reasonable transformation and writes the results to the destination array. For any element in the input that cannot be converted, `__dfault` is used instead.

`do_narrow()` is a hook for a derived facet to change the behavior of narrowing. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__dfault</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Implements [std::__ctype_abstract_base<_CharT>](#).

5.675.2.5 `template<typename _CharT> virtual const char_type* std::ctype<_CharT>::do_scan_is (mask __m, const char_type * __lo, const char_type * __hi) const` [protected], [virtual], [inherited]

Find char_type matching mask.

This function searches for and returns the first char_type c in [__lo,__hi) for which is(__m,c) is true.

do_scan_is() is a hook for a derived facet to change the behavior of match searching. do_is() must always return the same result for the same input.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to a matching char_type if found, else __hi.

Implements [std::__ctype_abstract_base<_CharT>](#).

5.675.2.6 `template<typename _CharT> virtual const char_type* std::ctype<_CharT>::do_scan_not (mask __m, const char_type * __lo, const char_type * __hi) const` [protected], [virtual], [inherited]

Find char_type not matching mask.

This function searches for and returns a pointer to the first char_type c of [lo,hi) for which is(m,c) is false.

do_scan_is() is a hook for a derived facet to change the behavior of match searching. do_is() must always return the same result for the same input.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to a non-matching `char_type` if found, else `__hi`.

Implements [std::__ctype_abstract_base<_CharT>](#).

5.675.2.7 `template<typename _CharT> virtual char_type std::ctype<_CharT>::do_tolower (char_type __c) const`
`[protected], [virtual], [inherited]`

Convert to lowercase.

This virtual function converts the argument to lowercase if possible. If not possible (for example, '2'), returns the argument.

`do_tolower()` is a hook for a derived facet to change the behavior of lowercasing. `do_tolower()` must always return the same result for the same input.

Parameters

<code>__c</code>	The <code>char_type</code> to convert.
------------------	--

Returns

The lowercase `char_type` if convertible, else `__c`.

Implements [std::__ctype_abstract_base<_CharT>](#).

5.675.2.8 `template<typename _CharT> virtual const char_type* std::ctype<_CharT>::do_tolower (char_type * __lo, const char_type * __hi) const`
`[protected], [virtual], [inherited]`

Convert array to lowercase.

This virtual function converts each `char_type` in the range `[__lo, __hi)` to lowercase if possible. Other elements remain untouched.

`do_tolower()` is a hook for a derived facet to change the behavior of lowercasing. `do_tolower()` must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns`__hi`.Implements [std::__ctype_abstract_base<_CharT>](#).

5.675.2.9 `template<typename _CharT> virtual char_type std::ctype<_CharT>::do_toupper (char_type __c) const`
`[protected], [virtual], [inherited]`

Convert to uppercase.

This virtual function converts the `char_type` argument to uppercase if possible. If not possible (for example, '2'), returns the argument.

`do_toupper()` is a hook for a derived facet to change the behavior of uppercasing. `do_toupper()` must always return the same result for the same input.

Parameters

<code>__c</code>	The <code>char_type</code> to convert.
------------------	--

ReturnsThe uppercase `char_type` if convertible, else `__c`.Implements [std::__ctype_abstract_base<_CharT>](#).

5.675.2.10 `template<typename _CharT> virtual const char_type* std::ctype<_CharT>::do_toupper (char_type * __lo, const char_type * __hi) const`
`[protected], [virtual], [inherited]`

Convert array to uppercase.

This virtual function converts each `char_type` in the range `[__lo, __hi)` to uppercase if possible. Other elements remain untouched.

`do_toupper()` is a hook for a derived facet to change the behavior of uppercasing. `do_toupper()` must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

Implements [std::__ctype_abstract_base<_CharT>](#).

5.675.2.11 `template<typename _CharT> virtual char_type std::ctype<_CharT>::do_widen (char __c) const`
`[protected], [virtual], [inherited]`

Widen char.

This virtual function converts the char to char_type using the simplest reasonable transformation.

do_widen() is a hook for a derived facet to change the behavior of widening. do_widen() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__c</code>	The char to convert.
------------------	----------------------

Returns

The converted char_type

Implements [std::__ctype_abstract_base<_CharT>](#).

5.675.2.12 `template<typename _CharT> virtual const char* std::ctype<_CharT>::do_widen (const char * __lo, const char * __hi, char_type * __to) const`
`[protected], [virtual], [inherited]`

Widen char array.

This function converts each char in the input to char_type using the simplest reasonable transformation.

do_widen() is a hook for a derived facet to change the behavior of widening. do_widen() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__lo</code>	Pointer to start range.
<code>__hi</code>	Pointer to end of range.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Implements [std::__ctype_abstract_base<_CharT>](#).

5.675.2.13 `template<typename _CharT> bool std::__ctype_abstract_base<_CharT>::is(mask __m, char_type __c) const [inline],[inherited]`

Test `char_type` classification.

This function finds a mask `M` for `__c` and compares it to mask `__m`. It does so by returning the value of `ctype<char_type>::do_is()`.

Parameters

<code>__c</code>	The <code>char_type</code> to compare the mask of.
<code>__m</code>	The mask to compare against.

Returns

`(M & __m) != 0`.

Definition at line 169 of file `locale_facets.h`.

Referenced by `std::time_get<_CharT, _InIter>::get()`, `std::ctype<char>::is()`, `std::isalnum()`, `std::isalpha()`, `std::isblank()`, `std::isctrl()`, `std::isdigit()`, `std::isgraph()`, `std::islower()`, `std::isprint()`, `std::ispunct()`, `std::isspace()`, `std::isupper()`, `std::isxdigit()`, and `std::basic_istream<_CharT, _Traits>::sentry::sentry()`.

5.675.2.14 `template<typename _CharT> const char_type* std::__ctype_abstract_base<_CharT>::is(const char_type * __lo, const char_type * __hi, mask * __vec) const [inline],[inherited]`

Return a mask array.

This function finds the mask for each `char_type` in the range `[lo,hi)` and successively writes it to `vec`. `vec` must have as many elements as the char array. It does so by returning the value of `ctype<char_type>::do_is()`.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__vec</code>	Pointer to an array of mask storage.

Returns

`__hi`.

Definition at line 186 of file locale_facets.h.

5.675.2.15 `template<typename _CharT> char std::__ctype_abstract_base<_CharT>::narrow(char_type __c, char __default) const [inline],[inherited]`

Narrow char_type to char.

This function converts the char_type to char using the simplest reasonable transformation. If the conversion fails, default is returned instead. It does so by returning ctype<char_type>::do_narrow(__c).

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__c</code>	The char_type to convert.
<code>__default</code>	Char to return if conversion fails.

Returns

The converted char.

Definition at line 331 of file locale_facets.h.

Referenced by std::time_get<_CharT, _InIter>::do_date_order(), std::time_get<_CharT, _InIter>::get(), and std::time_put<_CharT, _OutIter>::put().

5.675.2.16 `template<typename _CharT> const char_type* std::__ctype_abstract_base<_CharT>::narrow(const char_type * __lo, const char_type * __hi, char __default, char * __to) const [inline],[inherited]`

Narrow array to char array.

This function converts each char_type in the input to char using the simplest reasonable transformation and writes the results to the destination array. For any char_type in the input that cannot be converted, default is used instead. It does so by returning ctype<char_type>::do_narrow(__lo, __hi, __default, __to).

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__default</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Definition at line 353 of file locale_facets.h.

5.675.2.17 `template<typename _CharT> const char_type* std::__ctype_abstract_base<_CharT>::scan_is(mask __m, const char_type* __lo, const char_type* __hi) const` `[inline]`, `[inherited]`

Find char_type matching a mask.

This function searches for and returns the first char_type c in [lo,hi) for which is(m,c) is true. It does so by returning ctype<char_type>::do_scan_is().

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to matching char_type if found, else `__hi`.

Definition at line 202 of file locale_facets.h.

Referenced by std::ctype<char>::is().

5.675.2.18 `template<typename _CharT> const char_type* std::__ctype_abstract_base<_CharT>::scan_not(mask __m, const char_type* __lo, const char_type* __hi) const` `[inline]`, `[inherited]`

Find char_type not matching a mask.

This function searches for and returns the first char_type c in [lo,hi) for which is(m,c) is false. It does so by returning ctype<char_type>::do_scan_not().

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to first char in range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to non-matching char if found, else `__hi`.

Definition at line 218 of file locale_facets.h.

Referenced by std::money_get<_CharT, _InIter>::do_get(), and std::ctype<char>::scan_is().

5.675.2.19 `template<typename _CharT> char_type std::__ctype_abstract_base<_CharT>::tolower (char_type __c)`
`const [inline],[inherited]`

Convert to lowercase.

This function converts the argument to lowercase if possible. If not possible (for example, '2'), returns the argument. It does so by returning `ctype<char_type>::do_tolower(c)`.

Parameters

<code>__c</code>	The char_type to convert.
------------------	---------------------------

Returns

The lowercase char_type if convertible, else `__c`.

Definition at line 261 of file `locale_facets.h`.

Referenced by `std::time_get<_CharT, _InIter>::get()`, and `std::tolower()`.

5.675.2.20 `template<typename _CharT> const char_type* std::__ctype_abstract_base<_CharT>::tolower (char_type`
`* __lo, const char_type * __hi) const [inline],[inherited]`

Convert array to lowercase.

This function converts each char_type in the range `[__lo,__hi)` to lowercase if possible. Other elements remain untouched. It does so by returning `ctype<char_type>::do_tolower(__lo, __hi)`.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

Definition at line 276 of file `locale_facets.h`.

5.675.2.21 `template<typename _CharT> char_type std::__ctype_abstract_base<_CharT>::toupper (char_type __c)`
`const [inline],[inherited]`

Convert to uppercase.

This function converts the argument to uppercase if possible. If not possible (for example, '2'), returns the argument. It does so by returning `ctype<char_type>::do_toupper()`.

Parameters

\longleftrightarrow _c	The char_type to convert.
-----------------------------	---------------------------

Returns

The uppercase char_type if convertible, else __c.

Definition at line 232 of file locale_facets.h.

Referenced by std::time_get< _CharT, _InIter >::do_date_order(), std::time_get< _CharT, _InIter >::get(), and std::toupper().

5.675.2.22 `template<typename _CharT> const char_type* std::__ctype_abstract_base< _CharT >::toupper (char_type * __lo, const char_type * __hi) const [inline],[inherited]`

Convert array to uppercase.

This function converts each char_type in the range [lo,hi) to uppercase if possible. Other elements remain untouched. It does so by returning ctype<char_type>::do_toupper(lo, hi).

Parameters

\longleftrightarrow __lo	Pointer to start of range.
\longleftrightarrow __hi	Pointer to end of range.

Returns

__hi.

Definition at line 247 of file locale_facets.h.

5.675.2.23 `template<typename _CharT> char_type std::__ctype_abstract_base< _CharT >::widen (char __c) const [inline],[inherited]`

Widen char to char_type.

This function converts the char argument to char_type using the simplest reasonable transformation. It does so by returning ctype<char_type>::do_widen(c).

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>_↔</code>	The char to convert.
<code>_c</code>	

Returns

The converted `char_type`.

Definition at line 293 of file `locale_facets.h`.

Referenced by `std::time_get<_CharT, _Inlter>::do_get()`, `std::money_get<_CharT, _Inlter>::do_get()`, `std::num_↔_get<_CharT, _Inlter>::do_get()`, `std::time_put<_CharT, _Outlter>::do_put()`, `std::money_put<_CharT, _Outlter>::do_put()`, and `std::num_put<_CharT, _Outlter>::do_put()`.

5.675.2.24 `template<typename _CharT> const char* std::__ctype_abstract_base<_CharT>::widen (const char * __lo, const char * __hi, char_type * __to) const` `[inline]`, `[inherited]`

Widen array to `char_type`.

This function converts each char in the input to `char_type` using the simplest reasonable transformation. It does so by returning `ctype<char_type>::do_widen(c)`.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>_↔</code> <code>_lo</code>	Pointer to start of range.
<code>_↔</code> <code>_hi</code>	Pointer to end of range.
<code>_↔</code> <code>_to</code>	Pointer to the destination array.

Returns

`__hi`.

Definition at line 312 of file `locale_facets.h`.

5.675.3 Member Data Documentation

5.675.3.1 `template<typename _CharT> locale::id std::ctype<_CharT>::id` `[static]`, `[inherited]`

The facet id for `ctype<char_type>`

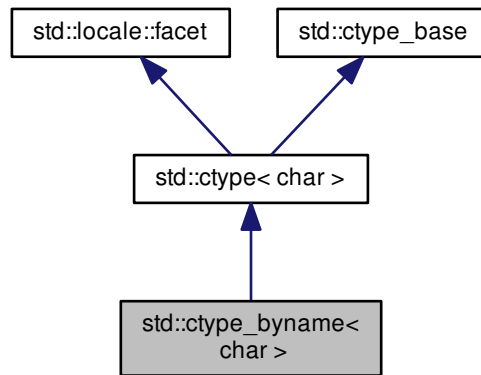
Definition at line 620 of file `locale_facets.h`.

The documentation for this class was generated from the following file:

- [locale_facets.h](#)

5.676 std::ctype_byname< char > Class Template Reference

Inheritance diagram for std::ctype_byname< char >:



Public Types

- typedef const int * **__to_type**
- typedef char [char_type](#)
- typedef unsigned short **mask**

Public Member Functions

- **ctype_byname** (const char * __s, size_t __refs=0)
- **ctype_byname** (const [string](#) & __s, size_t __refs=0)
- bool **is** (mask __m, char __c) const
- const char * **is** (const char * __lo, const char * __hi, mask * __vec) const
- char **narrow** ([char_type](#) __c, char __dfault) const
- const [char_type](#) * **narrow** (const [char_type](#) * __lo, const [char_type](#) * __hi, char __dfault, char * __to) const
- const char * **scan_is** (mask __m, const char * __lo, const char * __hi) const
- const char * **scan_not** (mask __m, const char * __lo, const char * __hi) const
- const mask * **table** () const throw ()
- [char_type](#) **tolower** ([char_type](#) __c) const
- const [char_type](#) * **tolower** ([char_type](#) * __lo, const [char_type](#) * __hi) const
- [char_type](#) **toupper** ([char_type](#) __c) const
- const [char_type](#) * **toupper** ([char_type](#) * __lo, const [char_type](#) * __hi) const
- [char_type](#) **widen** (char __c) const
- const char * **widen** (const char * __lo, const char * __hi, [char_type](#) * __to) const

Static Public Member Functions

- static const mask * [classic_table](#) () throw ()

Static Public Attributes

- static const mask **alnum**
- static const mask **alpha**
- static const mask **blank**
- static const mask **cntrl**
- static const mask **digit**
- static const mask **graph**
- static [locale::id](#) id
- static const mask **lower**
- static const mask **print**
- static const mask **punct**
- static const mask **space**
- static const size_t [table_size](#)
- static const mask **upper**
- static const mask **xdigit**

Protected Member Functions

- virtual char [do_narrow](#) (char_type __c, char __dfault) const
- virtual const char_type * [do_narrow](#) (const char_type * __lo, const char_type * __hi, char __dfault, char * __to) const
- virtual char_type [do_tolower](#) (char_type __c) const
- virtual const char_type * [do_tolower](#) (char_type * __lo, const char_type * __hi) const
- virtual char_type [do_toupper](#) (char_type __c) const
- virtual const char_type * [do_toupper](#) (char_type * __lo, const char_type * __hi) const
- virtual char_type [do_widen](#) (char __c) const
- virtual const char * [do_widen](#) (const char * __lo, const char * __hi, char_type * __to) const

Static Protected Member Functions

- static __c_locale [_S_clone_c_locale](#) (__c_locale & __cloc) throw ()
- static void [_S_create_c_locale](#) (__c_locale & __cloc, const char * __s, __c_locale __old=0)
- static void [_S_destroy_c_locale](#) (__c_locale & __cloc)
- static __c_locale [_S_get_c_locale](#) ()
- static const char * [_S_get_c_name](#) () throw ()
- static __c_locale [_S_lc_ctype_c_locale](#) (__c_locale __cloc, const char * __s)

Protected Attributes

- `__c_locale` **`_M_c_locale_ctype`**
- `bool` **`_M_del`**
- `char` **`_M_narrow`** [1+static_cast< unsigned char >(-1)]
- `char` **`_M_narrow_ok`**
- `const mask *` **`_M_table`**
- `__to_type` **`_M_tolower`**
- `__to_type` **`_M_toupper`**
- `char` **`_M_widen`** [1+static_cast< unsigned char >(-1)]
- `char` **`_M_widen_ok`**

5.676.1 Detailed Description

```
template<>
class std::ctype_byname< char >
```

22.2.1.4 Class `ctype_byname` specializations.

Definition at line 1495 of file `locale_facets.h`.

5.676.2 Member Typedef Documentation

5.676.2.1 `typedef char` **`std::ctype< char >::char_type`** [inherited]

Typedef for the template parameter `char`.

Definition at line 686 of file `locale_facets.h`.

5.676.3 Member Function Documentation

5.676.3.1 `static const mask*` **`std::ctype< char >::classic_table () throw`** [static],[inherited]

Returns a pointer to the C locale mask table.

5.676.3.2 `virtual char` **`std::ctype< char >::do_narrow (char_type __c, char __dfault) const`** [inline],[protected],[virtual],[inherited]

Narrow `char`.

This virtual function converts the `char` to `char` using the simplest reasonable transformation. If the conversion fails, `dfault` is returned instead. For an underived `ctype<char>` facet, `c` will be returned unchanged.

`do_narrow()` is a hook for a derived facet to change the behavior of narrowing. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The char to convert.
<code>__default</code>	Char to return if conversion fails.

Returns

The converted char.

Definition at line 1131 of file locale_facets.h.

5.676.3.3 `virtual const char_type* std::ctype< char >::do_narrow (const char_type * __lo, const char_type * __hi, char __default, char * __to) const` [inline], [protected], [virtual], [inherited]

Narrow char array to char array.

This virtual function converts each char in the range [lo,hi) to char using the simplest reasonable transformation and writes the results to the destination array. For any char in the input that cannot be converted, *default* is used instead. For an underived `ctype<char>` facet, the argument will be copied unchanged.

`do_narrow()` is a hook for a derived facet to change the behavior of narrowing. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__default</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Definition at line 1157 of file locale_facets.h.

5.676.3.4 `virtual char_type std::ctype< char >::do_tolower (char_type __c) const` [protected], [virtual], [inherited]

Convert to lowercase.

This virtual function converts the char argument to lowercase if possible. If not possible (for example, '2'), returns the argument.

`do_tolower()` is a hook for a derived facet to change the behavior of lowercasing. `do_tolower()` must always return the same result for the same input.

Parameters

\leftrightarrow _c	The char to convert.
-------------------------	----------------------

Returns

The lowercase char if convertible, else __c.

5.676.3.5 `virtual const char_type* std::ctype< char >::do_toupper (char_type * __lo, const char_type * __hi) const` `[protected]`, `[virtual]`, `[inherited]`

Convert array to lowercase.

This virtual function converts each char in the range [lo,hi) to lowercase if possible. Other chars remain untouched.

do_toupper() is a hook for a derived facet to change the behavior of lowercasing. do_toupper() must always return the same result for the same input.

Parameters

\leftrightarrow _lo	Pointer to first char in range.
\leftrightarrow _hi	Pointer to end of range.

Returns

__hi.

5.676.3.6 `virtual char_type std::ctype< char >::do_toupper (char_type __c) const` `[protected]`, `[virtual]`, `[inherited]`

Convert to uppercase.

This virtual function converts the char argument to uppercase if possible. If not possible (for example, '2'), returns the argument.

do_toupper() is a hook for a derived facet to change the behavior of uppercasing. do_toupper() must always return the same result for the same input.

Parameters

\leftrightarrow _c	The char to convert.
-------------------------	----------------------

Returns

The uppercase char if convertible, else `__c`.

5.676.3.7 `virtual const char_type* std::ctype< char >::do_toupper (char_type * __lo, const char_type * __hi) const` `[protected], [virtual], [inherited]`

Convert array to uppercase.

This virtual function converts each char in the range [lo,hi) to uppercase if possible. Other chars remain untouched.

`do_toupper()` is a hook for a derived facet to change the behavior of uppercasing. `do_toupper()` must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

5.676.3.8 `virtual char_type std::ctype< char >::do_widen (char __c) const` `[inline], [protected], [virtual], [inherited]`

Widen char.

This virtual function converts the char to char using the simplest reasonable transformation. For an underived `ctype<char>` facet, the argument will be returned unchanged.

`do_widen()` is a hook for a derived facet to change the behavior of widening. `do_widen()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The char to convert.
------------------	----------------------

Returns

The converted character.

Definition at line 1082 of file `locale_facets.h`.

5.676.3.9 `virtual const char* std::ctype< char >::do_widen (const char * __lo, const char * __hi, char_type * __to) const`
`[inline], [protected], [virtual], [inherited]`

Widen char array.

This function converts each char in the range [lo,hi) to char using the simplest reasonable transformation. For an undervied ctype<char> facet, the argument will be copied unchanged.

do_widen() is a hook for a derived facet to change the behavior of widening. do_widen() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Definition at line 1105 of file locale_facets.h.

5.676.3.10 `bool std::ctype< char >::is (mask __m, char __c) const` `[inline], [inherited]`

Test char classification.

This function compares the mask table[c] to `__m`.

Parameters

<code>__c</code>	The char to compare the mask of.
<code>__m</code>	The mask to compare against.

Returns

True if `__m & table[__c]` is true, false otherwise.

Definition at line 43 of file ctype_inline.h.

References `std::__ctype_abstract_base< _CharT >::is()`.

5.676.3.11 `const char * std::ctype< char >::is (const char * __lo, const char * __hi, mask * __vec) const` `[inline]`, `[inherited]`

Return a mask array.

This function finds the mask for each char in the range [lo, hi) and successively writes it to vec. vec must have as many elements as the char array.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__vec</code>	Pointer to an array of mask storage.

Returns

`__hi`.

Definition at line 48 of file `ctype_inline.h`.

References `std::__ctype_abstract_base< _CharT >::scan_is()`.

5.676.3.12 `char std::ctype< char >::narrow (char_type __c, char __dfault) const` `[inline]`, `[inherited]`

Narrow char.

This function converts the char to char using the simplest reasonable transformation. If the conversion fails, dfault is returned instead. For an underived `ctype<char>` facet, c will be returned unchanged.

This function works as if it returns `ctype<char>::do_narrow(c)`. `do_narrow()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The char to convert.
<code>__dfault</code>	Char to return if conversion fails.

Returns

The converted character.

Definition at line 930 of file `locale_facets.h`.

References `std::__ctype_abstract_base< _CharT >::do_narrow()`.

5.676.3.13 `const char_type* std::ctype< char >::narrow (const char_type * __lo, const char_type * __hi, char __default, char * __to) const` `[inline], [inherited]`

Narrow char array.

This function converts each char in the input to char using the simplest reasonable transformation and writes the results to the destination array. For any char in the input that cannot be converted, *default* is used instead. For an underived ctype<char> facet, the argument will be copied unchanged.

This function works as if it returns ctype<char>::do_narrow(lo, hi, default, to). do_narrow() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.
<code>__default</code>	Char to use if conversion fails.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Definition at line 963 of file locale_facets.h.

References std::__ctype_abstract_base< _CharT >::do_narrow().

5.676.3.14 `const char * std::ctype< char >::scan_is (mask __m, const char * __lo, const char * __hi) const` `[inline], [inherited]`

Find char matching a mask.

This function searches for and returns the first char in [lo,hi) for which is(m,char) is true.

Parameters

<code>__m</code>	The mask to compare against.
<code>__lo</code>	Pointer to start of range.
<code>__hi</code>	Pointer to end of range.

Returns

Pointer to a matching char if found, else `__hi`.

Definition at line 57 of file ctype_inline.h.

References `std::ctype_abstract_base<_CharT>::scan_not()`.

5.676.3.15 `const char * std::ctype<char>::scan_not (mask __m, const char * __lo, const char * __hi) const` `[inline]`, `[inherited]`

Find char not matching a mask.

This function searches for and returns a pointer to the first char in `[__lo,__hi)` for which `is(m,char)` is false.

Parameters

<code>↔ __m</code>	The mask to compare against.
<code>↔ __lo</code>	Pointer to start of range.
<code>↔ __hi</code>	Pointer to end of range.

Returns

Pointer to a non-matching char if found, else `__hi`.

Definition at line 67 of file ctype_inline.h.

5.676.3.16 `const mask* std::ctype<char>::table () const throw` `[inline]`, `[inherited]`

Returns a pointer to the mask table provided to the constructor, or the default from `classic_table()` if none was provided.

Definition at line 981 of file locale_facets.h.

References `std::ctype_abstract_base<_CharT>::do_tolower()`, and `std::ctype_abstract_base<_CharT>::do←
_toupper()`.

5.676.3.17 `char_type std::ctype<char>::tolower (char_type __c) const` `[inline]`, `[inherited]`

Convert to lowercase.

This function converts the char argument to lowercase if possible. If not possible (for example, '2'), returns the argument.

`tolower()` acts as if it returns `ctype<char>::do_tolower(__c)`. `do_tolower()` must always return the same result for the same input.

Parameters

<code>↔ __c</code>	The char to convert.
------------------------	----------------------

Returns

The lowercase char if convertible, else `__c`.

Definition at line 835 of file locale_facets.h.

References `std::__ctype_abstract_base< _CharT >::do_tolower()`.

5.676.3.18 `const char_type* std::ctype< char >::tolower (char_type * __lo, const char_type * __hi) const`
`[inline],[inherited]`

Convert array to lowercase.

This function converts each char in the range [lo,hi) to lowercase if possible. Other chars remain untouched.

`tolower()` acts as if it returns `ctype<char>::do_tolower(__lo, __hi)`. `do_tolower()` must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to first char in range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

Definition at line 852 of file locale_facets.h.

References `std::__ctype_abstract_base< _CharT >::do_tolower()`.

5.676.3.19 `char_type std::ctype< char >::toupper (char_type __c) const` `[inline],[inherited]`

Convert to uppercase.

This function converts the char argument to uppercase if possible. If not possible (for example, '2'), returns the argument.

`toupper()` acts as if it returns `ctype<char>::do_toupper(c)`. `do_toupper()` must always return the same result for the same input.

Parameters

<code>__c</code>	The char to convert.
------------------	----------------------

Returns

The uppercase char if convertible, else `__c`.

Definition at line 802 of file `locale_facets.h`.

References `std::__ctype_abstract_base<_CharT>::do_toupper()`.

5.676.3.20 `const char_type* std::ctype<char>::toupper (char_type * __lo, const char_type * __hi) const`
`[inline],[inherited]`

Convert array to uppercase.

This function converts each char in the range `[__lo,__hi)` to uppercase if possible. Other chars remain untouched.

`toupper()` acts as if it returns `ctype<char>::do_toupper(__lo, __hi)`. `do_toupper()` must always return the same result for the same input.

Parameters

<code>__lo</code>	Pointer to first char in range.
<code>__hi</code>	Pointer to end of range.

Returns

`__hi`.

Definition at line 819 of file `locale_facets.h`.

References `std::__ctype_abstract_base<_CharT>::do_toupper()`.

5.676.3.21 `char_type std::ctype<char>::widen (char __c) const` `[inline],[inherited]`

Widen char.

This function converts the `char` to `char_type` using the simplest reasonable transformation. For an underived `ctype<char>` facet, the argument will be returned unchanged.

This function works as if it returns `ctype<char>::do_widen(c)`. `do_widen()` must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See `codecvt` for that.

Parameters

<code>__c</code>	The char to convert.
------------------	----------------------

Returns

The converted character.

Definition at line 872 of file locale_facets.h.

References std::__ctype_abstract_base< _CharT >::do_widen().

5.676.3.22 `const char* std::ctype< char >::widen (const char * __lo, const char * __hi, char_type * __to) const`
`[inline], [inherited]`

Widen char array.

This function converts each char in the input to char using the simplest reasonable transformation. For an underived ctype<char> facet, the argument will be copied unchanged.

This function works as if it returns ctype<char>::do_widen(c). do_widen() must always return the same result for the same input.

Note: this is not what you want for codepage conversions. See codecvt for that.

Parameters

<code>__lo</code>	Pointer to first char in range.
<code>__hi</code>	Pointer to end of range.
<code>__to</code>	Pointer to the destination array.

Returns

`__hi`.

Definition at line 899 of file locale_facets.h.

References std::__ctype_abstract_base< _CharT >::do_widen().

5.676.4 Member Data Documentation

5.676.4.1 `locale::id std::ctype< char >::id` `[static], [inherited]`

The facet id for ctype<char>

Definition at line 703 of file locale_facets.h.

5.676.4.2 `const size_t std::ctype< char >::table_size` `[static], [inherited]`

The size of the mask table. It is `SCHAR_MAX + 1`.

Definition at line 705 of file `locale_facets.h`.

The documentation for this class was generated from the following file:

- [locale_facets.h](#)

5.677 `std::decimal::decimal128` Class Reference

Public Types

- `typedef float __decfloat128 __attribute__((mode(TD)))`

Public Member Functions

- `decimal128` ([decimal32](#) d32)
- `decimal128` ([decimal64](#) d64)
- `decimal128` (float __r)
- `decimal128` (double __r)
- `decimal128` (long double __r)
- `decimal128` (int __z)
- `decimal128` (unsigned int __z)
- `decimal128` (long __z)
- `decimal128` (unsigned long __z)
- `decimal128` (long long __z)
- `decimal128` (unsigned long long __z)
- [decimal128](#) (__decfloat128 __z)
- `__decfloat128 __getval` (void)
- `void __setval` (__decfloat128 __x)
- `operator long long` () const
- [decimal128](#) & `operator*=` ([decimal32](#) __rhs)
- [decimal128](#) & `operator*=` ([decimal64](#) __rhs)
- [decimal128](#) & `operator*=` ([decimal128](#) __rhs)
- [decimal128](#) & `operator*=` (int __rhs)
- [decimal128](#) & `operator*=` (unsigned int __rhs)
- [decimal128](#) & `operator*=` (long __rhs)
- [decimal128](#) & `operator*=` (long long __rhs)
- [decimal128](#) & `operator*=` (unsigned long long __rhs)
- [decimal128](#) & `operator*=` (unsigned long __rhs)
- [decimal128](#) & `operator++` ()
- [decimal128](#) `operator++` (int)
- [decimal128](#) & `operator+=` ([decimal32](#) __rhs)
- [decimal128](#) & `operator+=` ([decimal64](#) __rhs)
- [decimal128](#) & `operator+=` (unsigned int __rhs)
- [decimal128](#) & `operator+=` (unsigned long long __rhs)

- [decimal128](#) & **operator+=** (long __rhs)
- [decimal128](#) & **operator+=** (unsigned long __rhs)
- [decimal128](#) & **operator+=** (long long __rhs)
- [decimal128](#) & **operator+=** (int __rhs)
- [decimal128](#) & **operator+=** ([decimal128](#) __rhs)
- [decimal128](#) & **operator--** ()
- [decimal128](#) **operator--** (int)
- [decimal128](#) & **operator-=** (int __rhs)
- [decimal128](#) & **operator-=** ([decimal32](#) __rhs)
- [decimal128](#) & **operator-=** (unsigned long __rhs)
- [decimal128](#) & **operator-=** ([decimal64](#) __rhs)
- [decimal128](#) & **operator-=** (long long __rhs)
- [decimal128](#) & **operator-=** (long __rhs)
- [decimal128](#) & **operator-=** ([decimal128](#) __rhs)
- [decimal128](#) & **operator-=** (unsigned int __rhs)
- [decimal128](#) & **operator-=** (unsigned long long __rhs)
- [decimal128](#) & **operator/=** ([decimal64](#) __rhs)
- [decimal128](#) & **operator/=** (unsigned long __rhs)
- [decimal128](#) & **operator/=** (unsigned long long __rhs)
- [decimal128](#) & **operator/=** (long __rhs)
- [decimal128](#) & **operator/=** (long long __rhs)
- [decimal128](#) & **operator/=** (int __rhs)
- [decimal128](#) & **operator/=** ([decimal128](#) __rhs)
- [decimal128](#) & **operator/=** (unsigned int __rhs)
- [decimal128](#) & **operator/=** ([decimal32](#) __rhs)

5.677.1 Detailed Description

3.2.4 Class decimal128.

Definition at line 399 of file decimal.

5.677.2 Constructor & Destructor Documentation

5.677.2.1 std::decimal::decimal128::decimal128 (__decfloat128 __z) [inline]

Conforming extension: Conversion from scalar decimal type.

Definition at line 424 of file decimal.

The documentation for this class was generated from the following file:

- [decimal](#)

5.678 std::decimal::decimal32 Class Reference

Public Types

- typedef float __decfloat32 **__attribute__**((mode(SD)))

Public Member Functions

- **decimal32** ([decimal64](#) __d64)
- **decimal32** ([decimal128](#) __d128)
- **decimal32** (float __r)
- **decimal32** (double __r)
- **decimal32** (long double __r)
- **decimal32** (int __z)
- **decimal32** (unsigned int __z)
- **decimal32** (long __z)
- **decimal32** (unsigned long __z)
- **decimal32** (long long __z)
- **decimal32** (unsigned long long __z)
- [decimal32](#) (__decfloat32 __z)
- __decfloat32 **__getval** (void)
- void **__setval** (__decfloat32 __x)
- **operator long long** () const
- [decimal32](#) & **operator*=** ([decimal32](#) __rhs)
- [decimal32](#) & **operator*=** ([decimal64](#) __rhs)
- [decimal32](#) & **operator*=** ([decimal128](#) __rhs)
- [decimal32](#) & **operator*=** (int __rhs)
- [decimal32](#) & **operator*=** (unsigned int __rhs)
- [decimal32](#) & **operator*=** (long __rhs)
- [decimal32](#) & **operator*=** (long long __rhs)
- [decimal32](#) & **operator*=** (unsigned long long __rhs)
- [decimal32](#) & **operator*=** (unsigned long __rhs)
- [decimal32](#) & **operator++** ()
- [decimal32](#) **operator++** (int)
- [decimal32](#) & **operator+=** ([decimal32](#) __rhs)
- [decimal32](#) & **operator+=** ([decimal64](#) __rhs)
- [decimal32](#) & **operator+=** (unsigned int __rhs)
- [decimal32](#) & **operator+=** (unsigned long long __rhs)
- [decimal32](#) & **operator+=** (long __rhs)
- [decimal32](#) & **operator+=** (unsigned long __rhs)
- [decimal32](#) & **operator+=** (long long __rhs)
- [decimal32](#) & **operator+=** (int __rhs)
- [decimal32](#) & **operator+=** ([decimal128](#) __rhs)
- [decimal32](#) & **operator--** ()
- [decimal32](#) **operator--** (int)
- [decimal32](#) & **operator-=** (int __rhs)
- [decimal32](#) & **operator-=** ([decimal32](#) __rhs)
- [decimal32](#) & **operator-=** (unsigned long __rhs)
- [decimal32](#) & **operator-=** ([decimal64](#) __rhs)
- [decimal32](#) & **operator-=** (long long __rhs)

- [decimal32](#) & **operator=** (long __rhs)
- [decimal32](#) & **operator=** ([decimal128](#) __rhs)
- [decimal32](#) & **operator=** (unsigned int __rhs)
- [decimal32](#) & **operator=** (unsigned long long __rhs)
- [decimal32](#) & **operator/=** ([decimal64](#) __rhs)
- [decimal32](#) & **operator/=** (unsigned long __rhs)
- [decimal32](#) & **operator/=** (unsigned long long __rhs)
- [decimal32](#) & **operator/=** (long __rhs)
- [decimal32](#) & **operator/=** (long long __rhs)
- [decimal32](#) & **operator/=** (int __rhs)
- [decimal32](#) & **operator/=** ([decimal128](#) __rhs)
- [decimal32](#) & **operator/=** (unsigned int __rhs)
- [decimal32](#) & **operator/=** ([decimal32](#) __rhs)

5.678.1 Detailed Description

3.2.2 Class decimal32.

Definition at line 227 of file decimal.

5.678.2 Constructor & Destructor Documentation

5.678.2.1 std::decimal::decimal32::decimal32 (__decfloat32 __z) `[inline]`

Conforming extension: Conversion from scalar decimal type.

Definition at line 251 of file decimal.

The documentation for this class was generated from the following file:

- [decimal](#)

5.679 std::decimal::decimal64 Class Reference

Public Types

- typedef float __decfloat64 __attribute__((mode(DD)))

Public Member Functions

- **decimal64** ([decimal32](#) d32)
- **decimal64** ([decimal128](#) d128)
- **decimal64** (float __r)
- **decimal64** (double __r)
- **decimal64** (long double __r)
- **decimal64** (int __z)
- **decimal64** (unsigned int __z)
- **decimal64** (long __z)
- **decimal64** (unsigned long __z)
- **decimal64** (long long __z)
- **decimal64** (unsigned long long __z)
- [decimal64](#) (__decfloat64 __z)
- [__decfloat64](#) **__getval** (void)
- void **__setval** (__decfloat64 __x)
- **operator long long** () const
- [decimal64](#) & **operator*=** ([decimal32](#) __rhs)
- [decimal64](#) & **operator*=** ([decimal64](#) __rhs)
- [decimal64](#) & **operator*=** ([decimal128](#) __rhs)
- [decimal64](#) & **operator*=** (int __rhs)
- [decimal64](#) & **operator*=** (unsigned int __rhs)
- [decimal64](#) & **operator*=** (long __rhs)
- [decimal64](#) & **operator*=** (long long __rhs)
- [decimal64](#) & **operator*=** (unsigned long long __rhs)
- [decimal64](#) & **operator*=** (unsigned long __rhs)
- [decimal64](#) & **operator++** ()
- [decimal64](#) **operator++** (int)
- [decimal64](#) & **operator+=** ([decimal32](#) __rhs)
- [decimal64](#) & **operator+=** ([decimal64](#) __rhs)
- [decimal64](#) & **operator+=** (unsigned int __rhs)
- [decimal64](#) & **operator+=** (unsigned long long __rhs)
- [decimal64](#) & **operator+=** (long __rhs)
- [decimal64](#) & **operator+=** (unsigned long __rhs)
- [decimal64](#) & **operator+=** (long long __rhs)
- [decimal64](#) & **operator+=** (int __rhs)
- [decimal64](#) & **operator+=** ([decimal128](#) __rhs)
- [decimal64](#) & **operator--** ()
- [decimal64](#) **operator--** (int)
- [decimal64](#) & **operator-=** (int __rhs)
- [decimal64](#) & **operator-=** ([decimal32](#) __rhs)
- [decimal64](#) & **operator-=** (unsigned long __rhs)
- [decimal64](#) & **operator-=** ([decimal64](#) __rhs)
- [decimal64](#) & **operator-=** (long long __rhs)
- [decimal64](#) & **operator-=** (long __rhs)
- [decimal64](#) & **operator-=** ([decimal128](#) __rhs)
- [decimal64](#) & **operator-=** (unsigned int __rhs)
- [decimal64](#) & **operator-=** (unsigned long long __rhs)
- [decimal64](#) & **operator/=** ([decimal64](#) __rhs)
- [decimal64](#) & **operator/=** (unsigned long __rhs)
- [decimal64](#) & **operator/=** (unsigned long long __rhs)

- [decimal64](#) & **operator**/= (long __rhs)
- [decimal64](#) & **operator**/= (long long __rhs)
- [decimal64](#) & **operator**/= (int __rhs)
- [decimal64](#) & **operator**/= ([decimal128](#) __rhs)
- [decimal64](#) & **operator**/= (unsigned int __rhs)
- [decimal64](#) & **operator**/= ([decimal32](#) __rhs)

5.679.1 Detailed Description

3.2.3 Class decimal64.

Definition at line 313 of file decimal.

5.679.2 Constructor & Destructor Documentation

5.679.2.1 `std::decimal::decimal64::decimal64 (__decfloat64 __z) [inline]`

Conforming extension: Conversion from scalar decimal type.

Definition at line 337 of file decimal.

The documentation for this class was generated from the following file:

- [decimal](#)

5.680 `std::default_delete<_Tp>` Struct Template Reference

Public Member Functions

- constexpr [default_delete](#) () noexcept=default
- template<typename _Up, typename = typename enable_if<is_convertible<_Up*, _Tp*>::value>::type> [default_delete](#) (const [default_delete](#)<_Up> &) noexcept
- void [operator\(\)](#) (_Tp *__ptr) const

5.680.1 Detailed Description

```
template<typename _Tp>
struct std::default_delete<_Tp>
```

Primary template of `default_delete`, used by `unique_ptr`.

Definition at line 54 of file `unique_ptr.h`.

5.680.2 Constructor & Destructor Documentation

5.680.2.1 `template<typename _Tp> constexpr std::default_delete<_Tp>::default_delete () [default],
[noexcept]`

Default constructor.

5.680.2.2 `template<typename _Tp> template<typename _Up, typename = typename enable_if<is_convertible<_Up*,
_Tp*>::value>::type> std::default_delete<_Tp>::default_delete (const default_delete<_Up> &)
[inline], [noexcept]`

Converting constructor.

Allows conversion from a deleter for arrays of another type, `_Up`, only if `_Up*` is convertible to `_Tp*`.

Definition at line 66 of file `unique_ptr.h`.

5.680.3 Member Function Documentation

5.680.3.1 `template<typename _Tp> void std::default_delete<_Tp>::operator() (_Tp * __ptr) const [inline]`

Calls `delete __ptr`.

Definition at line 70 of file `unique_ptr.h`.

The documentation for this struct was generated from the following file:

- [unique_ptr.h](#)

5.681 `std::default_delete<_Tp[]>` Struct Template Reference

Public Member Functions

- constexpr [default_delete](#) () noexcept=default
- template<typename _Up, typename = typename enable_if<is_convertible<_Up(*)[], _Tp(*)[]>::value>::type> [default_delete](#) (const [default_delete](#)<_Up[]> &) noexcept
- template<typename _Up> enable_if< is_convertible< _Up(*)[], _Tp(*)[]>::value >::type [operator\(\)](#) (_Up * __ptr) const

5.681.1 Detailed Description

```
template<typename _Tp>
struct std::default_delete<_Tp[]>
```

Specialization for arrays, `default_delete`.

Definition at line 84 of file `unique_ptr.h`.

5.681.2 Constructor & Destructor Documentation

5.681.2.1 `template<typename _Tp> constexpr std::default_delete<_Tp[]>::default_delete () [default],
[noexcept]`

Default constructor.

5.681.2.2 `template<typename _Tp> template<typename _Up, typename = typename enable_if<is_convertible<_Up(*)[],
_Tp(*)[]>::value>::type> std::default_delete<_Tp[]>::default_delete (const default_delete<_Up[]> &)
[inline], [noexcept]`

Converting constructor.

Allows conversion from a deleter for arrays of another type, such as a const-qualified version of `_Tp`.

Conversions from types derived from `_Tp` are not allowed because it is unsafe to `delete[]` an array of derived types through a pointer to the base type.

Definition at line 101 of file `unique_ptr.h`.

5.681.3 Member Function Documentation

5.681.3.1 `template<typename _Tp> template<typename _Up> enable_if<is_convertible<_Up(*)[], _Tp(*)[]>::value>::type
std::default_delete<_Tp[]>::operator() (_Up * __ptr) const [inline]`

Calls `delete[] __ptr`.

Definition at line 106 of file `unique_ptr.h`.

The documentation for this struct was generated from the following file:

- [unique_ptr.h](#)

5.682 std::defer_lock_t Struct Reference

5.682.1 Detailed Description

Do not acquire ownership of the mutex.

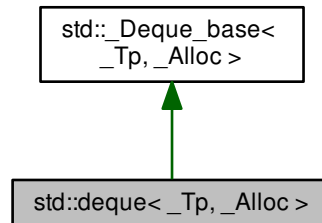
Definition at line 132 of file `std_mutex.h`.

The documentation for this struct was generated from the following file:

- [std_mutex.h](#)

5.683 `std::deque<_Tp, _Alloc>` Class Template Reference

Inheritance diagram for `std::deque<_Tp, _Alloc>`:



Public Types

- typedef `_Alloc` **allocator_type**
- typedef `_Base::const_iterator` **const_iterator**
- typedef `_Alloc_traits::const_pointer` **const_pointer**
- typedef `_Alloc_traits::const_reference` **const_reference**
- typedef `std::reverse_iterator< const_iterator >` **const_reverse_iterator**
- typedef `ptrdiff_t` **difference_type**
- typedef `_Base::iterator` **iterator**
- typedef `_Alloc_traits::pointer` **pointer**
- typedef `_Alloc_traits::reference` **reference**
- typedef `std::reverse_iterator< iterator >` **reverse_iterator**
- typedef `size_t` **size_type**
- typedef `_Tp` **value_type**

Public Member Functions

- `deque()`
- `deque(const allocator_type &__a)`
- `deque(size_type __n, const allocator_type &__a=allocator_type())`
- `deque(size_type __n, const value_type &__value, const allocator_type &__a=allocator_type())`
- `deque(const deque &__x)`
- `deque(deque &&__x)`
- `deque(const deque &__x, const allocator_type &__a)`
- `deque(deque &&__x, const allocator_type &__a)`
- `deque(initializer_list< value_type > __l, const allocator_type &__a=allocator_type())`
- `template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>>`
`deque(_InputIterator __first, _InputIterator __last, const allocator_type &__a=allocator_type())`
- `~deque()`
- `void assign(size_type __n, const value_type &__val)`

- template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>>
void **assign** (_InputIterator __first, _InputIterator __last)
- void **assign** (initializer_list< value_type > __l)
- reference **at** (size_type __n)
- const_reference **at** (size_type __n) const
- reference **back** () noexcept
- const_reference **back** () const noexcept
- iterator **begin** () noexcept
- const_iterator **begin** () const noexcept
- const_iterator **cbegin** () const noexcept
- const_iterator **cend** () const noexcept
- void **clear** () noexcept
- const_reverse_iterator **crbegin** () const noexcept
- const_reverse_iterator **crend** () const noexcept
- template<typename... _Args>
iterator **emplace** (const_iterator __position, _Args &&... __args)
- template<typename... _Args>
void **emplace_back** (_Args &&... __args)
- template<typename... _Args>
void **emplace_front** (_Args &&... __args)
- bool **empty** () const noexcept
- iterator **end** () noexcept
- const_iterator **end** () const noexcept
- iterator **erase** (const_iterator __position)
- iterator **erase** (const_iterator __first, const_iterator __last)
- reference **front** () noexcept
- const_reference **front** () const noexcept
- allocator_type **get_allocator** () const noexcept
- iterator **insert** (const_iterator __position, const value_type &__x)
- iterator **insert** (const_iterator __position, value_type &&__x)
- iterator **insert** (const_iterator __p, initializer_list< value_type > __l)
- iterator **insert** (const_iterator __position, size_type __n, const value_type &__x)
- template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>>
iterator **insert** (const_iterator __position, _InputIterator __first, _InputIterator __last)
- size_type **max_size** () const noexcept
- deque & **operator=** (const deque &__x)
- deque & **operator=** (deque &&__x) noexcept(_Alloc_traits::_S_always_equal())
- deque & **operator=** (initializer_list< value_type > __l)
- reference **operator[]** (size_type __n) noexcept
- const_reference **operator[]** (size_type __n) const noexcept
- void **pop_back** () noexcept
- void **pop_front** () noexcept
- void **push_back** (const value_type &__x)
- void **push_back** (value_type &&__x)
- void **push_front** (const value_type &__x)
- void **push_front** (value_type &&__x)
- reverse_iterator **rbegin** () noexcept
- const_reverse_iterator **rbegin** () const noexcept
- reverse_iterator **rend** () noexcept
- const_reverse_iterator **rend** () const noexcept
- void **resize** (size_type __new_size)

- void [resize](#) (size_type __new_size, const value_type &__x)
- void [shrink_to_fit](#) () noexcept
- size_type [size](#) () const noexcept
- void [swap](#) (deque &__x) noexcept

Protected Types

- enum { **_S_initial_map_size** }
- typedef [__gnu_cxx::__alloc_traits](#)< _Map_alloc_type > **_Map_alloc_traits**
- typedef _Alloc_traits::template rebind< _Ptr >::other **_Map_alloc_type**
- typedef _Alloc_traits::pointer **_Ptr**
- typedef _Alloc_traits::const_pointer **_Ptr_const**

Protected Member Functions

- _Map_pointer **_M_allocate_map** (size_t __n)
- _Ptr **_M_allocate_node** ()
- template<typename _InputIterator >
void **_M_assign_aux** (_InputIterator __first, _InputIterator __last, [std::input_iterator_tag](#))
- template<typename _ForwardIterator >
void **_M_assign_aux** (_ForwardIterator __first, _ForwardIterator __last, [std::forward_iterator_tag](#))
- template<typename _Integer >
void **_M_assign_dispatch** (_Integer __n, _Integer __val, __true_type)
- template<typename _InputIterator >
void **_M_assign_dispatch** (_InputIterator __first, _InputIterator __last, __false_type)
- void **_M_create_nodes** (_Map_pointer __nstart, _Map_pointer __nfinish)
- void **_M_deallocate_map** (_Map_pointer __p, size_t __n) noexcept
- void **_M_deallocate_node** (_Ptr __p) noexcept
- void **_M_default_append** (size_type __n)
- void **_M_default_initialize** ()
- template<typename _Alloc1 >
void **_M_destroy_data** (iterator __first, iterator __last, const _Alloc1 &)
- void **_M_destroy_data** (iterator __first, iterator __last, const [std::allocator](#)< _Tp > &)
- void **_M_destroy_data_aux** (iterator __first, iterator __last)
- void **_M_destroy_nodes** (_Map_pointer __nstart, _Map_pointer __nfinish) noexcept
- iterator **_M_erase** (iterator __pos)
- iterator **_M_erase** (iterator __first, iterator __last)
- void **_M_erase_at_begin** (iterator __pos)
- void **_M_erase_at_end** (iterator __pos)
- void **_M_fill_assign** (size_type __n, const value_type &__val)
- void **_M_fill_initialize** (const value_type &__value)
- void **_M_fill_insert** (iterator __pos, size_type __n, const value_type &__x)
- _Map_alloc_type **_M_get_map_allocator** () const noexcept
- _Tp_alloc_type & **_M_get_Tp_allocator** () noexcept
- const _Tp_alloc_type & **_M_get_Tp_allocator** () const noexcept
- template<typename _Integer >
void **_M_initialize_dispatch** (_Integer __n, _Integer __x, __true_type)
- template<typename _InputIterator >
void **_M_initialize_dispatch** (_InputIterator __first, _InputIterator __last, __false_type)

- void [_M_initialize_map](#) (size_t)
 - template<typename... _Args>
[iterator](#) [_M_insert_aux](#) ([iterator](#) __pos, _Args &&... __args)
 - void [_M_insert_aux](#) ([iterator](#) __pos, size_type __n, const value_type &__x)
 - template<typename _ForwardIterator >
void [_M_insert_aux](#) ([iterator](#) __pos, _ForwardIterator __first, _ForwardIterator __last, size_type __n)
 - template<typename _Integer >
void [_M_insert_dispatch](#) ([iterator](#) __pos, _Integer __n, _Integer __x, __true_type)
 - template<typename _InputIterator >
void [_M_insert_dispatch](#) ([iterator](#) __pos, _InputIterator __first, _InputIterator __last, __false_type)
 - void [_M_move_assign1](#) ([deque](#) &&__x, [true_type](#)) noexcept
 - void [_M_move_assign1](#) ([deque](#) &&__x, [false_type](#))
 - void [_M_move_assign2](#) ([deque](#) &&__x, [true_type](#))
 - void [_M_move_assign2](#) ([deque](#) &&__x, [false_type](#))
 - void [_M_range_check](#) (size_type __n) const
 - template<typename _InputIterator >
void [_M_range_insert_aux](#) ([iterator](#) __pos, _InputIterator __first, _InputIterator __last, [std::input_iterator_tag](#))
 - template<typename _ForwardIterator >
void [_M_range_insert_aux](#) ([iterator](#) __pos, _ForwardIterator __first, _ForwardIterator __last, [std::forward_iterator_tag](#))
 - template<typename... _Args>
void [_M_replace_map](#) (_Args &&... __args)
 - bool [_M_shrink_to_fit](#) ()
-
- template<typename _InputIterator >
void [_M_range_initialize](#) (_InputIterator __first, _InputIterator __last, [std::input_iterator_tag](#))
 - template<typename _ForwardIterator >
void [_M_range_initialize](#) (_ForwardIterator __first, _ForwardIterator __last, [std::forward_iterator_tag](#))
-
- template<typename... _Args>
void [_M_push_back_aux](#) (_Args &&... __args)
 - template<typename... _Args>
void [_M_push_front_aux](#) (_Args &&... __args)
 - void [_M_pop_back_aux](#) ()
 - void [_M_pop_front_aux](#) ()
-
- [iterator](#) [_M_reserve_elements_at_front](#) (size_type __n)
 - [iterator](#) [_M_reserve_elements_at_back](#) (size_type __n)
 - void [_M_new_elements_at_front](#) (size_type __new_elements)
 - void [_M_new_elements_at_back](#) (size_type __new_elements)
-
- void [_M_reserve_map_at_back](#) (size_type __nodes_to_add=1)
 - void [_M_reserve_map_at_front](#) (size_type __nodes_to_add=1)
 - void [_M_reallocate_map](#) (size_type __nodes_to_add, bool __add_at_front)

Static Protected Member Functions

- static size_t [_S_buffer_size](#) () noexcept

Protected Attributes

- `_Deque_impl _M_impl`

5.683.1 Detailed Description

```
template<typename _Tp, typename _Alloc = std::allocator<_Tp>>
class std::deque<_Tp, _Alloc>
```

A standard container using fixed-size memory allocation and constant-time manipulation of elements at either end.

Template Parameters

<code>_Tp</code>	Type of element.
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_Tp></code> .

Meets the requirements of a [container](#), a [reversible container](#), and a [sequence](#), including the [optional sequence requirements](#).

In previous HP/SGI versions of `deque`, there was an extra template parameter so users could control the node size. This extension turned out to violate the C++ standard (it can be detected using template template parameters), and it was removed.

Here's how a `deque<Tp>` manages memory. Each deque has 4 members:

- `Tp** _M_map`
- `size_t _M_map_size`
- `iterator _M_start, _M_finish`

`map_size` is at least 8. `map` is an array of `map_size` pointers-to-*nodes*. (The name `map` has nothing to do with the `std::map` class, and **nodes** should not be confused with `std::list`'s usage of *node*.)

A *node* has no specific type name as such, but it is referred to as *node* in this file. It is a simple array-of-`Tp`. If `Tp` is very large, there will be one `Tp` element per node (i.e., an *array* of one). For non-huge `Tp`'s, node size is inversely related to `Tp` size: the larger the `Tp`, the fewer `Tp`'s will fit in a node. The goal here is to keep the total size of a node relatively small and constant over different `Tp`'s, to improve allocator efficiency.

Not every pointer in the `map` array will point to a node. If the initial number of elements in the deque is small, the /middle/ `map` pointers will be valid, and the ones at the edges will be unused. This same situation will arise as the `map` grows: available `map` pointers, if any, will be on the ends. As new nodes are created, only a subset of the `map`'s pointers need to be copied *outward*.

Class invariants:

- For any nonsingular iterator `i`:

- i.node points to a member of the map array. (Yes, you read that correctly: i.node does not actually point to a node.) The member of the map array is what actually points to the node.
 - i.first == *(i.node) (This points to the node (first Tp element).)
 - i.last == i.first + node_size
 - i.cur is a pointer in the range [i.first, i.last). NOTE: the implication of this is that i.cur is always a dereferenceable pointer, even if i is a past-the-end iterator.
- Start and Finish are always nonsingular iterators. NOTE: this means that an empty deque must have one node, a deque with <N elements (where N is the node buffer size) must have one node, a deque with N through (2N-1) elements must have two nodes, etc.
 - For every node other than start.node and finish.node, every element in the node is an initialized object. If start.↔node == finish.node, then [start.cur, finish.cur) are initialized objects, and the elements outside that range are uninitialized storage. Otherwise, [start.cur, start.last) and [finish.first, finish.cur) are initialized objects, and [start.↔first, start.cur) and [finish.cur, finish.last) are uninitialized storage.
 - [map, map + map_size) is a valid, non-empty range.
 - [start.node, finish.node] is a valid range contained within [map, map + map_size).
 - A pointer in the range [map, map + map_size) points to an allocated node if and only if the pointer is in the range [start.node, finish.node].

Here's the magic: nothing in deque is **aware** of the discontinuous storage!

The memory setup and layout occurs in the parent, _Base, and the iterator class is entirely responsible for *leaping* from one node to the next. All the implementation routines for deque itself work only through the start and finish iterators. This keeps the routines simple and sane, and we can use other standard algorithms as well.

Definition at line 829 of file stl_deque.h.

5.683.2 Constructor & Destructor Documentation

5.683.2.1 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::deque< _Tp, _Alloc >::deque ()`
`[inline]`

Creates a deque with no elements.

Definition at line 884 of file stl_deque.h.

5.683.2.2 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::deque< _Tp, _Alloc >::deque (const`
`allocator_type &__a) [inline], [explicit]`

Creates a deque with no elements.

Parameters

<code>__a</code>	An allocator object.
------------------	----------------------

Definition at line 891 of file stl_deque.h.

5.683.2.3 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::deque<_Tp, _Alloc>::deque (size_type __n, const allocator_type & __a = allocator_type()) [inline], [explicit]`

Creates a deque with default constructed elements.

Parameters

<code>__n</code>	The number of elements to initially create.
<code>__a</code>	An allocator.

This constructor fills the deque with *n* default constructed elements.

Definition at line 904 of file `stl_deque.h`.

5.683.2.4 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::deque<_Tp, _Alloc>::deque (size_type __n, const value_type & __value, const allocator_type & __a = allocator_type()) [inline]`

Creates a deque with copies of an exemplar element.

Parameters

<code>__n</code>	The number of elements to initially create.
<code>__value</code>	An element to copy.
<code>__a</code>	An allocator.

This constructor fills the deque with `__n` copies of `__value`.

Definition at line 916 of file `stl_deque.h`.

5.683.2.5 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::deque<_Tp, _Alloc>::deque (const deque<_Tp, _Alloc> & __x) [inline]`

Deque copy constructor.

Parameters

<code>__x</code>	A deque of identical element and allocator types.
------------------	---

The newly-created deque uses a copy of the allocation object used by `__x`.

Definition at line 943 of file `stl_deque.h`.

5.683.2.6 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::deque<_Tp, _Alloc>::deque (deque<_Tp, _Alloc> && __x) [inline]`

Deque move constructor.

Parameters

\leftrightarrow __x	A deque of identical element and allocator types.
--------------------------	---

The newly-created deque contains the exact contents of __x. The contents of __x are a valid, but unspecified deque.

Definition at line 958 of file stl_deque.h.

5.683.2.7 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::deque<_Tp, _Alloc>::deque (const deque<_Tp, _Alloc> &__x, const allocator_type &__a) [inline]`

Copy constructor with alternative allocator.

Definition at line 962 of file stl_deque.h.

5.683.2.8 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::deque<_Tp, _Alloc>::deque (deque<_Tp, _Alloc> &&__x, const allocator_type &__a) [inline]`

Move constructor with alternative allocator.

Definition at line 969 of file stl_deque.h.

5.683.2.9 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::deque<_Tp, _Alloc>::deque (initializer_list<value_type> __l, const allocator_type &__a = allocator_type()) [inline]`

Builds a deque from an initializer list.

Parameters

\leftrightarrow __l	An initializer_list.
\leftrightarrow __a	An allocator object.

Create a deque consisting of copies of the elements in the initializer_list __l.

This will call the element type's copy constructor N times (where N is __l.size()) and do no memory reallocation.

Definition at line 992 of file stl_deque.h.

5.683.2.10 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>> std::deque<_Tp, _Alloc>::deque (_InputIterator __first, _InputIterator __last, const allocator_type &__a = allocator_type()) [inline]`

Builds a deque from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__a</code>	An allocator object.

Create a deque consisting of copies of the elements from `[__first, __last)`.

If the iterators are forward, bidirectional, or random-access, then this will call the elements' copy constructor N times (where N is `distance(__first, __last)`) and do no memory reallocation. But if only input iterators are used, then this will do at most $2N$ calls to the copy constructor, and $\log N$ memory reallocations.

Definition at line 1019 of file `std_deque.h`.

```
5.683.2.11  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::deque< _Tp, _Alloc >::~~deque ( )
              [inline]
```

The dtor only erases the elements, and note that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1040 of file `std_deque.h`.

5.683.3 Member Function Documentation

```
5.683.3.1  template<typename _Tp , typename _Alloc > void deque::_M_fill_initialize ( const value_type & __value )
              [protected]
```

Fills the deque with copies of `value`.

Parameters

<code>__value</code>	Initial value.
----------------------	----------------

Returns

Nothing.

Precondition

`_M_start` and `_M_finish` have already been initialized, but none of the deque's elements have yet been constructed.

This function is called only when the user provides an explicit size (with or without an explicit exemplar value).

Definition at line 375 of file `deque.tcc`.

References `std::_Destroy()`, and `std::deque< _Tp, _Alloc >::_M_range_initialize()`.

Referenced by `std::deque< _Tp, _Alloc >::insert()`.

5.683.3.2 `template<typename _Tp, typename _Alloc > void std::_Deque_base< _Tp, _Alloc >::_M_initialize_map (size_t __num_elements)` [protected], [inherited]

Layout storage.

Parameters

<code>__num_elements</code>	The count of T's for which to allocate space at first.
-----------------------------	--

Returns

Nothing.

The initial underlying memory layout is a bit complicated...

Definition at line 680 of file `std_deque.h`.

References `std::max()`.

5.683.3.3 `template<typename _Tp, typename _Alloc > void deque::_M_new_elements_at_back (size_type __new_elements)` [protected]

Memory-handling helpers for the previous internal insert functions.

Definition at line 877 of file `deque.tcc`.

References `std::deque< _Tp, _Alloc >::_M_reallocate_map()`.

Referenced by `std::deque< _Tp, _Alloc >::_M_new_elements_at_front()`.

5.683.3.4 `template<typename _Tp, typename _Alloc > void deque::_M_new_elements_at_front (size_type __new_elements)` [protected]

Memory-handling helpers for the previous internal insert functions.

Definition at line 852 of file `deque.tcc`.

References `std::deque< _Tp, _Alloc >::_M_new_elements_at_back()`.

Referenced by `std::deque< _Tp, _Alloc >::_M_pop_front_aux()`.

5.683.3.5 `template<typename _Tp, typename _Alloc > void deque::_M_pop_back_aux ()` [protected]

Helper functions for `push_*` and `pop_*`.

Definition at line 531 of file `deque.tcc`.

References `std::deque< _Tp, _Alloc >::_M_pop_front_aux()`.

Referenced by `std::deque< _Tp, _Alloc >::_M_push_front_aux()`.

5.683.3.6 `template<typename _Tp, typename _Alloc > void deque::_M_pop_front_aux () [protected]`

Helper functions for `push_*` and `pop_*`.

Definition at line 547 of file `deque.tcc`.

References `std::_Destroy()`, `std::deque< _Tp, _Alloc >::_M_new_elements_at_front()`, `std::advance()`, `std::distance()`, and `std::inserter()`.

Referenced by `std::deque< _Tp, _Alloc >::_M_pop_back_aux()`.

5.683.3.7 `template<typename _Tp, typename _Alloc > template<typename... _Args> void deque::_M_push_back_aux (_Args &&... _args) [protected]`

Helper functions for `push_*` and `pop_*`.

Definition at line 463 of file `deque.tcc`.

References `std::deque< _Tp, _Alloc >::_M_push_front_aux()`.

Referenced by `std::deque< _Tp, _Alloc >::_M_range_initialize()`.

5.683.3.8 `template<typename _Tp, typename _Alloc > template<typename... _Args> void deque::_M_push_front_aux (_Args &&... _args) [protected]`

Helper functions for `push_*` and `pop_*`.

Definition at line 498 of file `deque.tcc`.

References `std::deque< _Tp, _Alloc >::_M_pop_back_aux()`.

Referenced by `std::deque< _Tp, _Alloc >::_M_push_back_aux()`.

5.683.3.9 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::deque< _Tp, _Alloc >::_M_range_check (size_type __n) const [inline], [protected]`

Safety check used only from `at()`.

Definition at line 1387 of file `stl_deque.h`.

5.683.3.10 `template<typename _Tp, typename _Alloc > template<typename _InputIterator > void deque::_M_range_initialize (_InputIterator __first, _InputIterator __last, std::input_iterator_tag) [protected]`

Fills the deque with whatever is in `[first,last)`.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Returns

Nothing.

If the iterators are actually forward iterators (or better), then the memory layout can be done all at once. Else we move forward using `push_back` on each value from the iterator.

Definition at line 401 of file `deque.tcc`.

Referenced by `std::deque< _Tp, _Alloc >::_M_fill_initialize()`.

5.683.3.11 `template<typename _Tp, typename _Alloc > template<typename _ForwardIterator > void deque::_M_range_initialize (_ForwardIterator __first, _ForwardIterator __last, std::forward_iterator_tag) [protected]`

Fills the deque with whatever is in `[first,last)`.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Returns

Nothing.

If the iterators are actually forward iterators (or better), then the memory layout can be done all at once. Else we move forward using `push_back` on each value from the iterator.

Definition at line 425 of file `deque.tcc`.

References `std::_Destroy()`, `std::deque< _Tp, _Alloc >::_M_push_back_aux()`, `std::advance()`, and `std::distance()`.

5.683.3.12 `template<typename _Tp, typename _Alloc > void deque::_M_reallocate_map (size_type __nodes_to_add, bool __add_at_front) [protected]`

Memory-handling helpers for the major map.

Makes sure the `_M_map` has space for new nodes. Does not actually add the nodes. Can invalidate `_M_map` pointers. (And consequently, deque iterators.)

Definition at line 902 of file `deque.tcc`.

References `std::max()`, and `std::min()`.

Referenced by `std::deque< _Tp, _Alloc >::_M_new_elements_at_back()`.

5.683.3.13 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> iterator std::deque< _Tp, _Alloc >::_M_reserve_elements_at_back (size_type __n) [inline], [protected]`

Memory-handling helpers for the previous internal insert functions.

Definition at line 2086 of file `stl_deque.h`.

5.683.3.14 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> iterator std::deque<_Tp, _Alloc>::M_reserve_elements_at_front(size_type __n) [inline], [protected]`

Memory-handling helpers for the previous internal insert functions.

Definition at line 2076 of file `stl_deque.h`.

5.683.3.15 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::deque<_Tp, _Alloc>::M_reserve_map_at_back(size_type __nodes_to_add = 1) [inline], [protected]`

Memory-handling helpers for the major map.

Makes sure the `_M_map` has space for new nodes. Does not actually add the nodes. Can invalidate `_M_map` pointers. (And consequently, deque iterators.)

Definition at line 2112 of file `stl_deque.h`.

5.683.3.16 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::deque<_Tp, _Alloc>::M_reserve_map_at_front(size_type __nodes_to_add = 1) [inline], [protected]`

Memory-handling helpers for the major map.

Makes sure the `_M_map` has space for new nodes. Does not actually add the nodes. Can invalidate `_M_map` pointers. (And consequently, deque iterators.)

Definition at line 2120 of file `stl_deque.h`.

5.683.3.17 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::deque<_Tp, _Alloc>::assign(size_type __n, const value_type & __val) [inline]`

Assigns a given value to a deque.

Parameters

<code>__n</code>	Number of elements to be assigned.
<code>__val</code>	Value to be assigned.

This function fills a deque with n copies of the given value. Note that the assignment completely changes the deque and that the resulting deque's size is the same as the number of elements assigned. Old data may be lost.

Definition at line 1100 of file `stl_deque.h`.

5.683.3.18 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> template<typename _InputIterator, typename = std::RequireInputIter<_InputIterator>> void std::deque<_Tp, _Alloc>::assign(_InputIterator __first, _InputIterator __last) [inline]`

Assigns a range to a deque.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

This function fills a deque with copies of the elements in the range `[__first,__last)`.

Note that the assignment completely changes the deque and that the resulting deque's size is the same as the number of elements assigned. Old data may be lost.

Definition at line 1119 of file `stl_deque.h`.

5.683.3.19 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::deque<_Tp, _Alloc >::assign (initializer_list<value_type> __l) [inline]`

Assigns an initializer list to a deque.

Parameters

<code>↔</code>	An initializer_list.
<code>__↔</code>	
<code>↔</code>	
<code>__↔</code>	
<code>l</code>	

This function fills a deque with copies of the elements in the initializer_list `__l`.

Note that the assignment completely changes the deque and that the resulting deque's size is the same as the number of elements assigned. Old data may be lost.

Definition at line 1144 of file `stl_deque.h`.

5.683.3.20 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> reference std::deque<_Tp, _Alloc >::at (size_type __n) [inline]`

Provides access to the data contained in the deque.

Parameters

<code>__↔</code>	The index of the element for which data should be accessed.
<code>__n</code>	

Returns

Read/write reference to data.

Exceptions

<code>std::out_of_range</code>	If <code>__n</code> is an invalid index.
--------------------------------	--

This function provides for safer data access. The parameter is first checked that it is in the range of the deque. The function throws `out_of_range` if the check fails.

Definition at line 1409 of file `stl_deque.h`.

5.683.3.21 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reference std::deque<_Tp, _Alloc>::at (size_type __n) const [inline]`

Provides access to the data contained in the deque.

Parameters

<code>__n</code>	The index of the element for which data should be accessed.
------------------	---

Returns

Read-only (constant) reference to data.

Exceptions

<code>std::out_of_range</code>	If <code>__n</code> is an invalid index.
--------------------------------	--

This function provides for safer data access. The parameter is first checked that it is in the range of the deque. The function throws `out_of_range` if the check fails.

Definition at line 1427 of file `stl_deque.h`.

5.683.3.22 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> reference std::deque<_Tp, _Alloc>::back () [inline], [noexcept]`

Returns a read/write reference to the data at the last element of the deque.

Definition at line 1454 of file `stl_deque.h`.

5.683.3.23 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reference std::deque<_Tp, _Alloc>::back () const [inline], [noexcept]`

Returns a read-only (constant) reference to the data at the last element of the deque.

Definition at line 1466 of file `stl_deque.h`.

5.683.3.24 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> iterator std::deque<_Tp, _Alloc>::begin ()`
`[inline], [noexcept]`

Returns a read/write iterator that points to the first element in the deque. Iteration is done in ordinary element order.

Definition at line 1159 of file `stl_deque.h`.

Referenced by `std::deque<_StateSeqT>::deque()`, `std::deque<_Tp, _Alloc>::operator=()`, and `std::operator==()`.

5.683.3.25 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_iterator std::deque<_Tp, _Alloc>::begin () const`
`[inline], [noexcept]`

Returns a read-only (constant) iterator that points to the first element in the deque. Iteration is done in ordinary element order.

Definition at line 1167 of file `stl_deque.h`.

5.683.3.26 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_iterator std::deque<_Tp, _Alloc>::cbegin () const`
`[inline], [noexcept]`

Returns a read-only (constant) iterator that points to the first element in the deque. Iteration is done in ordinary element order.

Definition at line 1230 of file `stl_deque.h`.

5.683.3.27 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_iterator std::deque<_Tp, _Alloc>::cend () const`
`[inline], [noexcept]`

Returns a read-only (constant) iterator that points one past the last element in the deque. Iteration is done in ordinary element order.

Definition at line 1239 of file `stl_deque.h`.

5.683.3.28 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::deque<_Tp, _Alloc>::clear ()`
`[inline], [noexcept]`

Erases all the elements. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1798 of file `stl_deque.h`.

5.683.3.29 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reverse_iterator std::deque<_Tp, _Alloc>::rbegin () const`
`[inline], [noexcept]`

Returns a read-only (constant) reverse iterator that points to the last element in the deque. Iteration is done in reverse element order.

Definition at line 1248 of file `stl_deque.h`.

5.683.3.30 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reverse_iterator std::deque<_Tp, _Alloc>::crend () const`
`[inline], [noexcept]`

Returns a read-only (constant) reverse iterator that points to one before the first element in the deque. Iteration is done in reverse element order.

Definition at line 1257 of file `stl_deque.h`.

5.683.3.31 `template<typename _Tp, typename _Alloc> template<typename... _Args> deque<_Tp, _Alloc>::iterator deque::emplace (const_iterator __position, _Args &&... __args)`

Inserts an object in deque before specified iterator.

Parameters

<code>__position</code>	A const_iterator into the deque.
<code>__args</code>	Arguments.

Returns

An iterator that points to the inserted data.

This function will insert an object of type T constructed with `T(std::forward<Args>(args)...) before the specified location.`

Definition at line 170 of file deque.tcc.

Referenced by `std::deque<_Tp, _Alloc>::operator=()`.

```
5.683.3.32  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> bool std::deque<_Tp, _Alloc>::empty ( )
            const [inline], [noexcept]
```

Returns true if the deque is empty. (Thus `begin()` would equal `end()`.)

Definition at line 1350 of file stl_deque.h.

```
5.683.3.33  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> iterator std::deque<_Tp, _Alloc>::end ( )
            [inline], [noexcept]
```

Returns a read/write iterator that points one past the last element in the deque. Iteration is done in ordinary element order.

Definition at line 1176 of file stl_deque.h.

Referenced by `std::deque<_StateSeqT>::deque()`, `std::deque<_Tp, _Alloc>::operator=()`, and `std::operator==()`.

```
5.683.3.34  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_iterator std::deque<_Tp, _Alloc>::end
            ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the deque. Iteration is done in ordinary element order.

Definition at line 1185 of file stl_deque.h.

```
5.683.3.35  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> iterator std::deque<_Tp, _Alloc>::erase (
            const_iterator __position ) [inline]
```

Remove element at given position.

Parameters

<code>__position</code>	Iterator pointing to element to be erased.
-------------------------	--

Returns

An iterator pointing to the next element (or end()).

This function will erase the element at the given position and thus shorten the deque by one.

The user is cautioned that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1744 of file stl_deque.h.

```
5.683.3.36  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> iterator std::deque<_Tp, _Alloc>::erase (
            const_iterator __first, const_iterator __last ) [inline]
```

Remove a range of elements.

Parameters

<code>__first</code>	Iterator pointing to the first element to be erased.
<code>__last</code>	Iterator pointing to one past the last element to be erased.

Returns

An iterator pointing to the element pointed to by *last* prior to erasing (or end()).

This function will erase the elements in the range [`__first`,`__last`) and shorten the deque accordingly.

The user is cautioned that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1768 of file stl_deque.h.

```
5.683.3.37  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> reference std::deque<_Tp, _Alloc>::front ( )
            [inline], [noexcept]
```

Returns a read/write reference to the data at the first element of the deque.

Definition at line 1438 of file stl_deque.h.

```
5.683.3.38  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reference std::deque<_Tp, _Alloc>
            >::front ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reference to the data at the first element of the deque.

Definition at line 1446 of file stl_deque.h.

5.683.3.39 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> allocator_type std::deque<_Tp, _Alloc>::get_allocator () const [inline], [noexcept]`

Get a copy of the memory allocation object.

Definition at line 1150 of file `stl_deque.h`.

Referenced by `std::deque<_Tp, _Alloc>::operator=()`.

5.683.3.40 `template<typename _Tp, typename _Alloc> deque<_Tp, _Alloc>::iterator deque::insert (const_iterator __position, const value_type & __x)`

Inserts given value into deque before specified iterator.

Parameters

<code>__position</code>	A const_iterator into the deque.
<code>__x</code>	Data to be inserted.

Returns

An iterator that points to the inserted data.

This function will insert a copy of the given value before the specified location.

Definition at line 194 of file deque.tcc.

References `std::deque<_Tp, _Alloc >::_M_fill_initialize()`, `std::begin()`, and `std::end()`.

5.683.3.41 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> iterator std::deque<_Tp, _Alloc >::insert (const_iterator __position, value_type && __x) [inline]`

Inserts given rvalue into deque before specified iterator.

Parameters

<code>__position</code>	A const_iterator into the deque.
<code>__x</code>	Data to be inserted.

Returns

An iterator that points to the inserted data.

This function will insert a copy of the given rvalue before the specified location.

Definition at line 1634 of file stl_deque.h.

5.683.3.42 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> iterator std::deque<_Tp, _Alloc >::insert (const_iterator __p, initializer_list<value_type> __l) [inline]`

Inserts an initializer list into the deque.

Parameters

<code>__p</code>	An iterator into the deque.
<code>__l</code>	An initializer_list.

This function will insert copies of the data in the initializer_list `__l` into the deque before the location specified by `__p`. This is known as *list insert*.

Definition at line 1647 of file `stl_deque.h`.

```
5.683.3.43  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> iterator std::deque<_Tp, _Alloc>::insert (
            const_iterator __position, size_type __n, const value_type & __x ) [inline]
```

Inserts a number of copies of given data into the deque.

Parameters

<code>__position</code>	A <code>const_iterator</code> into the deque.
<code>__n</code>	Number of elements to be inserted.
<code>__x</code>	Data to be inserted.

Returns

An iterator that points to the inserted data.

This function will insert a specified number of copies of the given data before the location specified by `__position`.

Definition at line 1663 of file `stl_deque.h`.

```
5.683.3.44  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> template<typename _InputIterator, typename =
            std::_RequireInputIter<_InputIterator>> iterator std::deque<_Tp, _Alloc>::insert ( const_iterator __position,
            _InputIterator __first, _InputIterator __last ) [inline]
```

Inserts a range into the deque.

Parameters

<code>__position</code>	A <code>const_iterator</code> into the deque.
<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Returns

An iterator that points to the inserted data.

This function will insert copies of the data in the range `[__first,__last)` into the deque before the location specified by `__position`. This is known as *range insert*.

Definition at line 1699 of file `stl_deque.h`.

```
5.683.3.45  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> size_type std::deque<_Tp, _Alloc>::max_size (
            ) const [inline],[noexcept]
```

Returns the `size()` of the largest possible deque.

Definition at line 1269 of file `stl_deque.h`.

5.683.3.46 `template<typename _Tp, typename _Alloc > deque< _Tp, _Alloc > & deque::operator= (const deque< _Tp, _Alloc > & __x)`

Deque assignment operator.

Parameters

<code>__x</code>	A deque of identical element and allocator types.
------------------	---

All the elements of `x` are copied, but unlike the copy constructor, the allocator object is not copied.

Definition at line 93 of file `deque.tcc`.

References `std::deque< _Tp, _Alloc >::begin()`, `std::deque< _Tp, _Alloc >::emplace()`, `std::deque< _Tp, _Alloc >::end()`, `std::deque< _Tp, _Alloc >::get_allocator()`, and `std::deque< _Tp, _Alloc >::size()`.

5.683.3.47 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> deque& std::deque< _Tp, _Alloc >::operator= (deque< _Tp, _Alloc > && __x) [inline], [noexcept]`

Deque move assignment operator.

Parameters

<code>__x</code>	A deque of identical element and allocator types.
------------------	---

The contents of `__x` are moved into this deque (without copying, if the allocators permit it). `__x` is a valid, but unspecified deque.

Definition at line 1063 of file `stl_deque.h`.

5.683.3.48 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> deque& std::deque< _Tp, _Alloc >::operator= (initializer_list< value_type > __l) [inline]`

Assigns an initializer list to a deque.

Parameters

<code>__l</code>	An <code>initializer_list</code> .
------------------	------------------------------------

This function fills a deque with copies of the elements in the `initializer_list __l`.

Note that the assignment completely changes the deque and that the resulting deque's size is the same as the number of elements assigned. Old data may be lost.

Definition at line 1082 of file `stl_deque.h`.

5.683.3.49 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> reference std::deque<_Tp, _Alloc>::operator[]
(size_type __n) [inline], [noexcept]`

Subscript access to the data contained in the deque.

Parameters

<code>__n</code>	The index of the element for which data should be accessed.
------------------	---

Returns

Read/write reference to data.

This operator allows for easy, array-style, data access. Note that data access with this operator is unchecked and `out_of_range` lookups are not defined. (For checked lookups see `at()`.)

Definition at line 1366 of file `stl_deque.h`.

5.683.3.50 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reference std::deque<_Tp, _Alloc>
>::operator[](size_type __n) const [inline], [noexcept]`

Subscript access to the data contained in the deque.

Parameters

<code>__n</code>	The index of the element for which data should be accessed.
------------------	---

Returns

Read-only (constant) reference to data.

This operator allows for easy, array-style, data access. Note that data access with this operator is unchecked and `out_of_range` lookups are not defined. (For checked lookups see `at()`.)

Definition at line 1381 of file `stl_deque.h`.

5.683.3.51 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::deque<_Tp, _Alloc>::pop_back ()
[inline], [noexcept]`

Removes last element.

This is a typical stack operation. It shrinks the deque by one.

Note that no data is returned, and if the last element's data is needed, it should be retrieved before `pop_back()` is called.

Definition at line 1571 of file `stl_deque.h`.

5.683.3.52 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::deque<_Tp, _Alloc >::pop_front ()`
`[inline], [noexcept]`

Removes first element.

This is a typical stack operation. It shrinks the deque by one.

Note that no data is returned, and if the first element's data is needed, it should be retrieved before `pop_front()` is called.

Definition at line 1549 of file `stl_deque.h`.

5.683.3.53 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::deque<_Tp, _Alloc >::push_back (`
`const value_type & __x) [inline]`

Add data to the end of the deque.

Parameters

<code>__x</code>	Data to be added.
------------------	-------------------

This is a typical stack operation. The function creates an element at the end of the deque and assigns the given data to it. Due to the nature of a deque this operation can be done in constant time.

Definition at line 1517 of file `stl_deque.h`.

5.683.3.54 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::deque<_Tp, _Alloc >::push_front (`
`const value_type & __x) [inline]`

Add data to the front of the deque.

Parameters

<code>__x</code>	Data to be added.
------------------	-------------------

This is a typical stack operation. The function creates an element at the front of the deque and assigns the given data to it. Due to the nature of a deque this operation can be done in constant time.

Definition at line 1484 of file `stl_deque.h`.

5.683.3.55 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> reverse_iterator std::deque<_Tp, _Alloc`
`>::rbegin () [inline], [noexcept]`

Returns a read/write reverse iterator that points to the last element in the deque. Iteration is done in reverse element order.

Definition at line 1194 of file `stl_deque.h`.

```
5.683.3.56  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reverse_iterator std::deque< _Tp,
            _Alloc >::rbegin ( ) const    [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last element in the deque. Iteration is done in reverse element order.

Definition at line 1203 of file stl_deque.h.

```
5.683.3.57  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> reverse_iterator std::deque< _Tp, _Alloc
            >::rend ( )    [inline], [noexcept]
```

Returns a read/write reverse iterator that points to one before the first element in the deque. Iteration is done in reverse element order.

Definition at line 1212 of file stl_deque.h.

```
5.683.3.58  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reverse_iterator std::deque< _Tp,
            _Alloc >::rend ( ) const    [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to one before the first element in the deque. Iteration is done in reverse element order.

Definition at line 1221 of file stl_deque.h.

```
5.683.3.59  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::deque< _Tp, _Alloc >::resize (
            size_type __new_size )    [inline]
```

Resizes the deque to the specified number of elements.

Parameters

<code>__new_size</code>	Number of elements the deque should contain.
-------------------------	--

This function will resize the deque to the specified number of elements. If the number is smaller than the deque's current size the deque is truncated, otherwise default constructed elements are appended.

Definition at line 1283 of file stl_deque.h.

```
5.683.3.60  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::deque< _Tp, _Alloc >::resize (
            size_type __new_size, const value_type &__x )    [inline]
```

Resizes the deque to the specified number of elements.

Parameters

<code>__new_size</code>	Number of elements the deque should contain.
<code>__x</code>	Data with which new elements should be populated.

This function will resize the deque to the specified number of elements. If the number is smaller than the deque's current size the deque is truncated, otherwise the deque is extended and new elements are populated with given data.

Definition at line 1305 of file `stl_deque.h`.

```
5.683.3.61 template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::deque<_Tp, _Alloc>::shrink_to_fit (
    ) [inline], [noexcept]
```

A non-binding request to reduce memory use.

Definition at line 1341 of file `stl_deque.h`.

```
5.683.3.62 template<typename _Tp, typename _Alloc = std::allocator<_Tp>> size_type std::deque<_Tp, _Alloc>::size ( )
    const [inline], [noexcept]
```

Returns the number of elements in the deque.

Definition at line 1264 of file `stl_deque.h`.

Referenced by `std::deque<_Tp, _Alloc>::operator=()`, and `std::operator==()`.

```
5.683.3.63 template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::deque<_Tp, _Alloc>::swap (
    deque<_Tp, _Alloc> &_x ) [inline], [noexcept]
```

Swaps data with another deque.

Parameters

<code>_↔</code>	A deque of the same element and allocator types.
<code>_x</code>	

This exchanges the elements between two deques in constant time. (Four pointers, so it should be quite fast.) Note that the global `std::swap()` function is specialized such that `std::swap(d1,d2)` will feed to this function.

Definition at line 1784 of file `stl_deque.h`.

The documentation for this class was generated from the following files:

- [stl_deque.h](#)
- [deque.tcc](#)

5.684 `std::discard_block_engine<_RandomNumberEngine, __p, __r>` Class Template Reference

Public Types

- typedef `_RandomNumberEngine::result_type` [result_type](#)

Public Member Functions

- [discard_block_engine](#) ()
- [discard_block_engine](#) (const _RandomNumberEngine &__rng)
- [discard_block_engine](#) (_RandomNumberEngine &&__rng)
- [discard_block_engine](#) (result_type __s)
- template<typename _Sseq , typename = typename std::enable_if<!std::is_same<_Sseq, discard_block_engine>::value && !std::is_←
same<_Sseq, _RandomNumberEngine>::value> ::type>
[discard_block_engine](#) (_Sseq &__q)
- const _RandomNumberEngine & [base](#) () const noexcept
- void [discard](#) (unsigned long long __z)
- [result_type operator](#)() ()
- void [seed](#) ()
- void [seed](#) (result_type __s)
- template<typename _Sseq >
void [seed](#) (_Sseq &__q)

Static Public Member Functions

- static constexpr [result_type max](#) ()
- static constexpr [result_type min](#) ()

Static Public Attributes

- static constexpr size_t **block_size**
- static constexpr size_t **used_block**

Friends

- template<typename _RandomNumberEngine1 , size_t __p1, size_t __r1, typename _CharT , typename _Traits >
[std::basic_ostream](#)< _CharT, _Traits > & [operator<<](#) ([std::basic_ostream](#)< _CharT, _Traits > &__os, const
[std::discard_block_engine](#)< _RandomNumberEngine1, __p1, __r1 > &__x)
- bool [operator==](#) (const [discard_block_engine](#) &__lhs, const [discard_block_engine](#) &__rhs)
- template<typename _RandomNumberEngine1 , size_t __p1, size_t __r1, typename _CharT , typename _Traits >
[std::basic_istream](#)< _CharT, _Traits > & [operator>>](#) ([std::basic_istream](#)< _CharT, _Traits > &__is, [std](#)←
::[discard_block_engine](#)< _RandomNumberEngine1, __p1, __r1 > &__x)

5.684.1 Detailed Description

```
template<typename _RandomNumberEngine, size_t __p, size_t __r>
class std::discard_block_engine< _RandomNumberEngine, __p, __r >
```

Produces random numbers from some base engine by discarding blocks of data.

0 <= __r <= __p

Definition at line 846 of file random.h.

5.684.2 Member Typedef Documentation

5.684.2.1 `template<typename _RandomNumberEngine, size_t __p, size_t __r> typedef _RandomNumberEngine::result_type
std::discard_block_engine<_RandomNumberEngine, __p, __r>::result_type`

The type of the generated random value.

Definition at line 849 of file random.h.

5.684.3 Constructor & Destructor Documentation

5.684.3.1 `template<typename _RandomNumberEngine, size_t __p, size_t __r> std::discard_block_engine<
_RandomNumberEngine, __p, __r>::discard_block_engine () [inline]`

Constructs a default discard_block_engine engine.

The underlying engine is default constructed as well.

Definition at line 864 of file random.h.

5.684.3.2 `template<typename _RandomNumberEngine, size_t __p, size_t __r> std::discard_block_engine<
_RandomNumberEngine, __p, __r>::discard_block_engine (const _RandomNumberEngine & __rng)
[inline], [explicit]`

Copy constructs a discard_block_engine engine.

Copies an existing base class random number generator.

Parameters

<code>__rng</code>	An existing (base class) engine object.
--------------------	---

Definition at line 874 of file random.h.

5.684.3.3 `template<typename _RandomNumberEngine, size_t __p, size_t __r> std::discard_block_engine<
_RandomNumberEngine, __p, __r>::discard_block_engine (_RandomNumberEngine && __rng) [inline],
[explicit]`

Move constructs a discard_block_engine engine.

Copies an existing base class random number generator.

Parameters

<code>__rng</code>	An existing (base class) engine object.
--------------------	---

Definition at line 884 of file random.h.

5.684.3.4 `template<typename _RandomNumberEngine, size_t __p, size_t __r> std::discard_block_engine<_RandomNumberEngine, __p, __r>::discard_block_engine(result_type __s) [inline],[explicit]`

Seed constructs a discard_block_engine engine.

Constructs the underlying generator engine seeded with __s.

Parameters

<code>__s</code>	A seed value for the base class engine.
------------------	---

Definition at line 894 of file random.h.

5.684.3.5 `template<typename _RandomNumberEngine, size_t __p, size_t __r> template<typename _Sseq, typename = typename std::enable_if<!std::is_same<_Sseq, discard_block_engine>::value && !std::is_same<_Sseq, _RandomNumberEngine>::value> ::type> std::discard_block_engine<_RandomNumberEngine, __p, __r>::discard_block_engine(_Sseq & __q) [inline],[explicit]`

Generator construct a discard_block_engine engine.

Parameters

<code>__q</code>	A seed sequence.
------------------	------------------

Definition at line 907 of file random.h.

5.684.4 Member Function Documentation

5.684.4.1 `template<typename _RandomNumberEngine, size_t __p, size_t __r> const _RandomNumberEngine& std::discard_block_engine<_RandomNumberEngine, __p, __r>::base() const [inline],[noexcept]`

Gets a const reference to the underlying generator engine object.

Definition at line 951 of file random.h.

Referenced by `std::shuffle_order_engine<_RandomNumberEngine, __k>::operator()()`.

5.684.4.2 `template<typename _RandomNumberEngine, size_t __p, size_t __r> void std::discard_block_engine<_RandomNumberEngine, __p, __r>::discard(unsigned long long __z) [inline]`

Discard a sequence of random numbers.

Definition at line 972 of file random.h.

5.684.4.3 `template<typename _RandomNumberEngine, size_t __p, size_t __r> static constexpr result_type
std::discard_block_engine<_RandomNumberEngine, __p, __r>::max() [inline], [static]`

Gets the maximum value in the generated random number range.

Definition at line 965 of file `random.h`.

References `std::max()`.

5.684.4.4 `template<typename _RandomNumberEngine, size_t __p, size_t __r> static constexpr result_type
std::discard_block_engine<_RandomNumberEngine, __p, __r>::min() [inline], [static]`

Gets the minimum value in the generated random number range.

Definition at line 958 of file `random.h`.

References `std::min()`.

5.684.4.5 `template<typename _RandomNumberEngine, size_t __p, size_t __r> discard_block_engine<
_RandomNumberEngine, __p, __r>::result_type std::discard_block_engine<_RandomNumberEngine, __p, __r
>::operator()()`

Gets the next value in the generated random number sequence.

Definition at line 688 of file `bits/random.tcc`.

References `std::dec()`, `std::fixed()`, `std::ios_base::flags()`, `std::left()`, `std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType>::operator>()()`, `std::__detail::operator>>()`, and `std::skipws()`.

Referenced by `std::subtract_with_carry_engine<_UIntType, __w, __s, __r>::operator>()()`.

5.684.4.6 `template<typename _RandomNumberEngine, size_t __p, size_t __r> void std::discard_block_engine<
_RandomNumberEngine, __p, __r>::seed() [inline]`

Reseeds the `discard_block_engine` object with the default seed for the underlying base class generator engine.

Definition at line 916 of file `random.h`.

5.684.4.7 `template<typename _RandomNumberEngine, size_t __p, size_t __r> void std::discard_block_engine<
_RandomNumberEngine, __p, __r>::seed(result_type __s) [inline]`

Reseeds the `discard_block_engine` object with the default seed for the underlying base class generator engine.

Definition at line 927 of file `random.h`.

5.684.4.8 `template<typename _RandomNumberEngine, size_t __p, size_t __r> template<typename _Sseq> void
std::discard_block_engine<_RandomNumberEngine, __p, __r>::seed(_Sseq & __q) [inline]`

Reseeds the `discard_block_engine` object with the given seed sequence.

Parameters

<code>__q</code>	A seed generator function.
------------------	----------------------------

Definition at line 940 of file random.h.

5.684.5 Friends And Related Function Documentation

5.684.5.1 `template<typename _RandomNumberEngine, size_t __p, size_t __r> template<typename _RandomNumberEngine1, size_t __p1, size_t __r1, typename _CharT, typename _Traits> std::basic_ostream<_CharT, _Traits>& operator<< (std::basic_ostream<_CharT, _Traits> & __os, const std::discard_block_engine<_RandomNumberEngine1, __p1, __r1> & __x) [friend]`

Inserts the current state of a `discard_block_engine` random number generator engine `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>discard_block_engine</code> random number generator engine.

Returns

The output stream with the state of `__x` inserted or in an error state.

5.684.5.2 `template<typename _RandomNumberEngine, size_t __p, size_t __r> bool operator== (const discard_block_engine<_RandomNumberEngine, __p, __r> & __lhs, const discard_block_engine<_RandomNumberEngine, __p, __r> & __rhs) [friend]`

Compares two `discard_block_engine` random number generator objects of the same type for equality.

Parameters

<code>__lhs</code>	A <code>discard_block_engine</code> random number generator object.
<code>__rhs</code>	Another <code>discard_block_engine</code> random number generator object.

Returns

true if the infinite sequences of generated values would be equal, false otherwise.

Definition at line 996 of file random.h.

```
5.684.5.3 template<typename _RandomNumberEngine, size_t __p, size_t __r> template<typename _RandomNumberEngine1,
size_t __p1, size_t __r1, typename _CharT, typename _Traits> std::basic_istream<_CharT, _Traits>& operator>> (
std::basic_istream<_CharT, _Traits> & __is, std::discard_block_engine<_RandomNumberEngine1, __p1,
__r1> & __x) [friend]
```

Extracts the current state of a % subtract_with_carry_engine random number generator engine __x from the input stream __is.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A discard_block_engine random number generator engine.

Returns

The input stream with the state of __x extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.685 std::discrete_distribution<_IntType> Class Template Reference

Classes

- struct [param_type](#)

Public Types

- typedef _IntType [result_type](#)

Public Member Functions

- template<typename _InputIterator> **discrete_distribution** (_InputIterator __wbegin, _InputIterator __wend)
- **discrete_distribution** ([initializer_list](#)< double > __wl)
- template<typename _Func> **discrete_distribution** (size_t __nw, double __xmin, double __xmax, _Func __fw)
- **discrete_distribution** (const [param_type](#) & __p)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator> void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator & __urng)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator> void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator & __urng, const [param_type](#) & __p)

- `template<typename _UniformRandomNumberGenerator >`
`void generate (result_type *__f, result_type *__t, _UniformRandomNumberGenerator &__urng, const param_type &__p)`
- `result_type max () const`
- `result_type min () const`
- `template<typename _UniformRandomNumberGenerator >`
`result_type operator() (_UniformRandomNumberGenerator &__urng)`
- `template<typename _UniformRandomNumberGenerator >`
`result_type operator() (_UniformRandomNumberGenerator &__urng, const param_type &__p)`
- `param_type param () const`
- `void param (const param_type &__param)`
- `std::vector< double > probabilities () const`
- `void reset ()`

Friends

- `template<typename _IntType1, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const std::discrete_distribution< _IntType1 > &__x)`
- `bool operator== (const discrete_distribution &__d1, const discrete_distribution &__d2)`
- `template<typename _IntType1, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, std::discrete_distribution< _IntType1 > &__x)`

5.685.1 Detailed Description

```
template<typename _IntType = int>
class std::discrete_distribution< _IntType >
```

A `discrete_distribution` random number distribution.

The formula for the discrete probability mass function is

Definition at line 5086 of file `random.h`.

5.685.2 Member Typedef Documentation

5.685.2.1 `template<typename _IntType = int> typedef _IntType std::discrete_distribution< _IntType >::result_type`

The type of the range of the distribution.

Definition at line 5089 of file `random.h`.

5.685.3 Member Function Documentation

5.685.3.1 `template<typename _IntType = int> result_type std::discrete_distribution<_IntType>::max () const`
`[inline]`

Returns the least upper bound value of the distribution.

Definition at line 5206 of file random.h.

5.685.3.2 `template<typename _IntType = int> result_type std::discrete_distribution<_IntType>::min () const`
`[inline]`

Returns the greatest lower bound value of the distribution.

Definition at line 5199 of file random.h.

5.685.3.3 `template<typename _IntType = int> template<typename _UniformRandomNumberGenerator> result_type`
`std::discrete_distribution<_IntType>::operator() (_UniformRandomNumberGenerator & __urng) [inline]`

Generating functions.

Definition at line 5217 of file random.h.

Referenced by `std::operator>>()`.

5.685.3.4 `template<typename _IntType = int> param_type std::discrete_distribution<_IntType>::param () const`
`[inline]`

Returns the parameter set of the distribution.

Definition at line 5184 of file random.h.

Referenced by `std::operator>>()`.

5.685.3.5 `template<typename _IntType = int> void std::discrete_distribution<_IntType>::param (const param_type &`
`__param) [inline]`

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

Definition at line 5192 of file random.h.

5.685.3.6 `template<typename _IntType = int> std::vector<double> std::discrete_distribution<_IntType>::probabilities (`
`) const [inline]`

Returns the probabilities of the distribution.

Definition at line 5174 of file random.h.

5.685.3.7 `template<typename _IntType = int> void std::discrete_distribution<_IntType>::reset() [inline]`

Resets the distribution state.

Definition at line 5167 of file random.h.

5.685.4 Friends And Related Function Documentation

5.685.4.1 `template<typename _IntType = int> template<typename _IntType1, typename _CharT, typename _Traits > std::basic_ostream<_CharT, _Traits>& operator<< (std::basic_ostream<_CharT, _Traits> & __os, const std::discrete_distribution<_IntType1> & __x) [friend]`

Inserts a discrete_distribution random number distribution __x into the output stream __os.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A discrete_distribution random number distribution.

Returns

The output stream with the state of __x inserted or in an error state.

5.685.4.2 `template<typename _IntType = int> bool operator==(const discrete_distribution<_IntType> & __d1, const discrete_distribution<_IntType> & __d2) [friend]`

Return true if two discrete distributions have the same parameters.

Definition at line 5252 of file random.h.

5.685.4.3 `template<typename _IntType = int> template<typename _IntType1, typename _CharT, typename _Traits > std::basic_istream<_CharT, _Traits>& operator>> (std::basic_istream<_CharT, _Traits> & __is, std::discrete_distribution<_IntType1> & __x) [friend]`

Extracts a discrete_distribution random number distribution __x from the input stream __is.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A discrete_distribution random number generator engine.

Returns

The input stream with `___x` extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.686 `std::discrete_distribution< _IntType >::param_type` Struct Reference

Public Types

- typedef [discrete_distribution< _IntType >](#) **distribution_type**

Public Member Functions

- template<typename _InputIterator >
param_type (_InputIterator __wbegin, _InputIterator __wend)
- **param_type** ([initializer_list< double >](#) __wil)
- template<typename _Func >
param_type (size_t __nw, double __xmin, double __xmax, _Func __fw)
- **param_type** (const [param_type](#) &)=default
- [param_type](#) & **operator=** (const [param_type](#) &)=default
- [std::vector< double >](#) **probabilities** () const

Friends

- class **discrete_distribution< _IntType >**
- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)

5.686.1 Detailed Description

```
template<typename _IntType = int>
struct std::discrete_distribution< _IntType >::param_type
```

Parameter type.

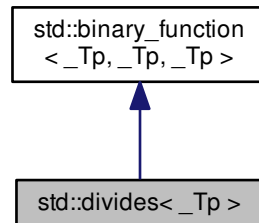
Definition at line 5095 of file random.h.

The documentation for this struct was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.687 `std::divides<_Tp>` Struct Template Reference

Inheritance diagram for `std::divides<_Tp>`:



Public Types

- typedef `_Tp` [first_argument_type](#)
- typedef `_Tp` [result_type](#)
- typedef `_Tp` [second_argument_type](#)

Public Member Functions

- `_GLIBCXX14_CONSTEXPR _Tp operator() (const _Tp &__x, const _Tp &__y) const`

5.687.1 Detailed Description

```
template<typename _Tp>
struct std::divides<_Tp>
```

One of the [math functors](#).

Definition at line 156 of file `stl_function.h`.

5.687.2 Member Typedef Documentation

5.687.2.1 typedef `_Tp` `std::binary_function<_Tp, _Tp, _Tp>::first_argument_type` [inherited]

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.687.2.2 `typedef _Tp std::binary_function<_Tp, _Tp, _Tp>::result_type` [inherited]

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.687.2.3 `typedef _Tp std::binary_function<_Tp, _Tp, _Tp>::second_argument_type` [inherited]

`second_argument_type` is the type of the second argument

Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.688 `std::divides< void >` Struct Template Reference

Public Types

- `typedef __is_transparent is_transparent`

Public Member Functions

- `template<typename _Tp, typename _Up>
_GLIBCXX14_CONSTEXPR auto operator() (_Tp &&__t, _Up &&__u) const noexcept(noexcept(std::forward<
_Tp>(__t)/std::forward<_Up>(__u))) -> decltype(std::forward<_Tp>(__t)/std::forward<_Up>(__u))`

5.688.1 Detailed Description

```
template<>
struct std::divides< void >
```

One of the [math functors](#).

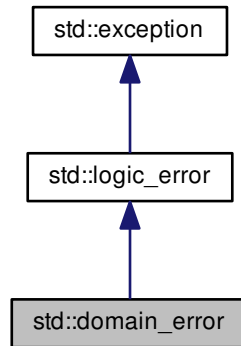
Definition at line 275 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.689 `std::domain_error` Class Reference

Inheritance diagram for `std::domain_error`:



Public Member Functions

- **`domain_error`** (const [string](#) &__arg) `_GLIBCXX_TXN_SAFE`
- **`domain_error`** (const char *) `_GLIBCXX_TXN_SAFE`
- virtual const char * [what](#) () const `_GLIBCXX_TXN_SAFE_DYN noexcept`

5.689.1 Detailed Description

Thrown by the library, or by you, to report domain errors (domain in the mathematical sense).

Definition at line 147 of file `stdexcept`.

5.689.2 Member Function Documentation

5.689.2.1 `virtual const char* std::logic_error::what () const` `[virtual]`, `[noexcept]`, `[inherited]`

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

Reimplemented in [std::future_error](#).

The documentation for this class was generated from the following file:

- [stdexcept](#)

5.690 std::enable_shared_from_this< _Tp > Class Template Reference

Public Member Functions

- [shared_ptr](#)< _Tp > **shared_from_this** ()
- [shared_ptr](#)< const _Tp > **shared_from_this** () const

Protected Member Functions

- **enable_shared_from_this** (const [enable_shared_from_this](#) &) noexcept
- [enable_shared_from_this](#) & **operator=** (const [enable_shared_from_this](#) &) noexcept

Friends

- `template<typename _Tp1, typename _Tp2 >
void __enable_shared_from_this_helper (const __shared_count<> &, const enable_shared_from_this< _Tp1 > *, const _Tp2 *) noexcept`

5.690.1 Detailed Description

```
template<typename _Tp>  
class std::enable_shared_from_this< _Tp >
```

Base class allowing use of member function `shared_from_this`.

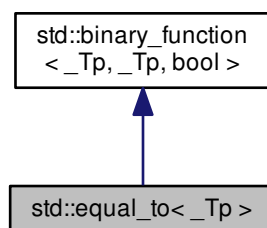
Definition at line 557 of file `bits/shared_ptr.h`.

The documentation for this class was generated from the following file:

- [bits/shared_ptr.h](#)

5.691 std::equal_to< _Tp > Struct Template Reference

Inheritance diagram for `std::equal_to< _Tp >`:



Public Types

- typedef `_Tp` [first_argument_type](#)
- typedef `bool` [result_type](#)
- typedef `_Tp` [second_argument_type](#)

Public Member Functions

- `_GLIBCXX14_CONSTEXPR` `bool` **operator()** (`const _Tp &__x`, `const _Tp &__y`) `const`

5.691.1 Detailed Description

```
template<typename _Tp>  
struct std::equal_to<_Tp>
```

One of the [comparison functors](#).

Definition at line 331 of file `stl_function.h`.

5.691.2 Member Typedef Documentation

5.691.2.1 `typedef _Tp std::binary_function<_Tp, _Tp, bool>::first_argument_type` `[inherited]`

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.691.2.2 `typedef bool std::binary_function<_Tp, _Tp, bool>::result_type` `[inherited]`

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.691.2.3 `typedef _Tp std::binary_function<_Tp, _Tp, bool>::second_argument_type` `[inherited]`

`second_argument_type` is the type of the second argument

Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.692 `std::equal_to< void >` Struct Template Reference

Public Types

- typedef `__is_transparent` is **`__transparent`**

Public Member Functions

- template<typename `_Tp` , typename `_Up` >
`_GLIBCXX14_CONSTEXPR` auto **`operator()`** (`_Tp` && `__t`, `_Up` && `__u`) const noexcept(noexcept(`std::forward`<
`_Tp` >(`__t`)==`std::forward`< `_Up` >(`__u`))) -> decltype(`std::forward`< `_Tp` >(`__t`)==`std::forward`< `_Up` >(`__u`))

5.692.1 Detailed Description

```
template<>
struct std::equal_to< void >
```

One of the [comparison functors](#).

Definition at line 412 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.693 `std::error_code` Struct Reference

Public Member Functions

- **`error_code`** (int `__v`, const `error_category` & `__cat`) noexcept
- template<typename `_ErrorCodeEnum` , typename = typename enable_if<is_error_code_enum< `_ErrorCodeEnum` >::value>::type>
`error_code` (`_ErrorCodeEnum` `__e`) noexcept
- void **`assign`** (int `__v`, const `error_category` & `__cat`) noexcept
- const `error_category` & **`category`** () const noexcept
- void **`clear`** () noexcept
- [error_condition](#) **`default_error_condition`** () const noexcept
- `_GLIBCXX_DEFAULT_ABI_TAG` [string](#) **`message`** () const
- **`operator bool`** () const noexcept
- template<typename `_ErrorCodeEnum` >
enable_if< [is_error_code_enum](#) < `_ErrorCodeEnum` >::value, [error_code](#) & >::type **`operator=`** (`_ErrorCodeEnum` `__e`) noexcept
- int **`value`** () const noexcept

Friends

- class **`hash< error_code >`**

5.693.1 Detailed Description

error_code

Definition at line 138 of file system_error.

The documentation for this struct was generated from the following file:

- [system_error](#)

5.694 std::error_condition Struct Reference

Public Member Functions

- **error_condition** (int __v, const error_category &__cat) noexcept
- template<typename _ErrorConditionEnum , typename = typename enable_if<is_error_condition_enum<_ErrorConditionEnum><↵
::value>::type>
error_condition (_ErrorConditionEnum __e) noexcept
- void **assign** (int __v, const error_category &__cat) noexcept
- const error_category & **category** () const noexcept
- void **clear** () noexcept
- _GLIBCXX_DEFAULT_ABI_TAG [string message](#) () const
- **operator bool** () const noexcept
- template<typename _ErrorConditionEnum >
enable_if< [is_error_condition_enum](#)<_ErrorConditionEnum>::value, [error_condition](#) & >::type **operator=** (_↵
ErrorConditionEnum __e) noexcept
- int **value** () const noexcept

5.694.1 Detailed Description

error_condition

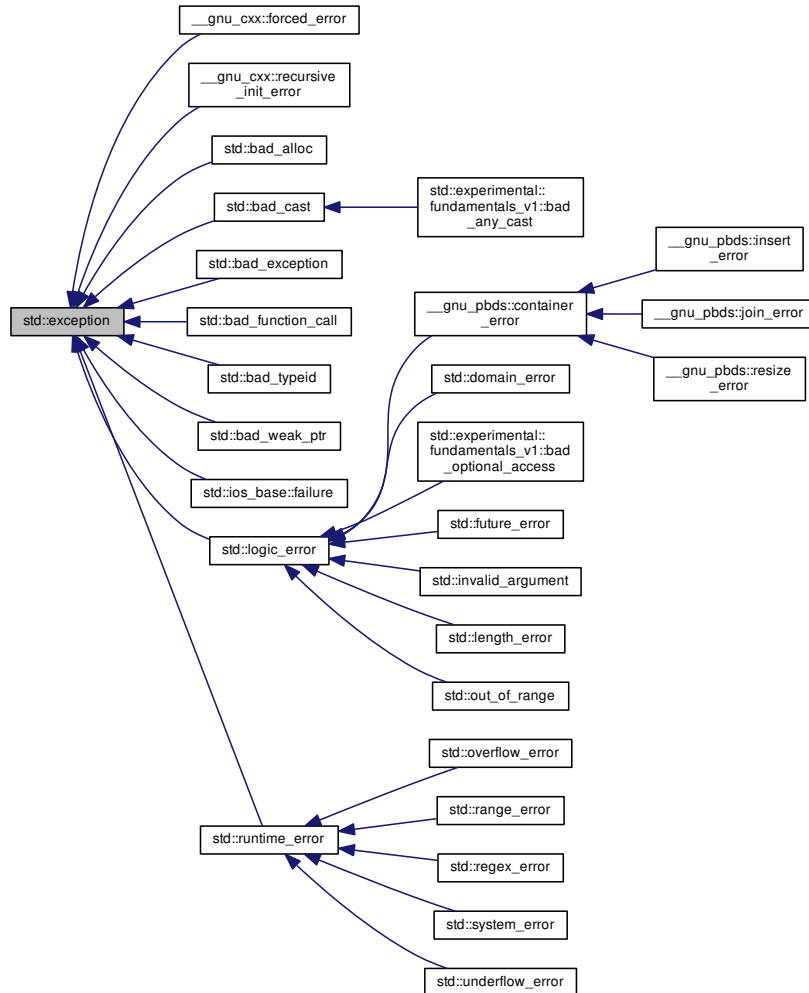
Definition at line 216 of file system_error.

The documentation for this struct was generated from the following file:

- [system_error](#)

5.695 std::exception Class Reference

Inheritance diagram for std::exception:



Public Member Functions

- virtual const char * [what](#) () const _GLIBCXX_TXN_SAFE_DYN noexcept

5.695.1 Detailed Description

Base class for all library exceptions.

This is the base class for all exceptions thrown by the standard library, and by certain language expressions. You are free to derive your own exception classes, or use a different hierarchy, or to throw non-class data (e.g., fundamental types).

Definition at line 60 of file exception.

5.695.2 Member Function Documentation

5.695.2.1 `virtual const char* std::exception::what () const` `[virtual], [noexcept]`

Returns a C-style character string describing the general cause of the current error.

Reimplemented in `std::bad_function_call`, `std::ios_base::failure`, `std::runtime_error`, `std::bad_typeid`, `std::bad_cast`, `std::logic_error`, `std::future_error`, `std::bad_exception`, `std::bad_weak_ptr`, `std::experimental::fundamentals_v1::bad_↵any_cast`, and `std::bad_alloc`.

The documentation for this class was generated from the following file:

- [exception](#)

5.696 `std::experimental::filesystem::v1::path` Class Reference

Inherited by `std::experimental::filesystem::v1::path::_Cmpt`.

Classes

- class [iterator](#)

Public Types

- typedef [iterator](#) **const_iterator**
- typedef `std::basic_string< value_type >` **string_type**
- typedef char **value_type**

Public Member Functions

- **path** (const [path](#) &__p)=default
- **path** ([path](#) &&__p) noexcept
- **path** ([string_type](#) &&__source)
- template<typename _Source , typename _Require = _Path<_Source>>
path (_Source const &__source)
- template<typename _InputIterator , typename _Require = _Path<_InputIterator, _InputIterator>>
path (_InputIterator __first, _InputIterator __last)
- template<typename _Source , typename _Require = _Path<_Source>, typename _Require2 = __value_type_is_char<_Source>>
path (_Source const &__source, const [locale](#) &__loc)
- template<typename _InputIterator , typename _Require = _Path<_InputIterator, _InputIterator>, typename _Require2 = __value_type_is_↵_char<_InputIterator>>
path (_InputIterator __first, _InputIterator __last, const [locale](#) &__loc)
- template<typename _Source >
_Path<_Source > & **append** (_Source const &__source)
- template<typename _InputIterator >
_Path<_InputIterator, _InputIterator > & **append** (_InputIterator __first, _InputIterator __last)

- `template<typename _Source >`
`_Path< _Source > & assign (_Source const &__source)`
- `template<typename _InputIterator >`
`_Path< _InputIterator, _InputIterator > & assign (_InputIterator __first, _InputIterator __last)`
- `const value_type * c_str ()` `const noexcept`
- `void clear ()` `noexcept`
- `int compare (const path &__p)` `const noexcept`
- `template<typename _Source >`
`_Path< _Source > & concat (_Source const &__x)`
- `template<typename _InputIterator >`
`_Path< _InputIterator, _InputIterator > & concat (_InputIterator __first, _InputIterator __last)`
- `bool empty ()` `const noexcept`
- `bool has_filename ()` `const`
- `bool has_parent_path ()` `const`
- `bool has_relative_path ()` `const`
- `bool has_root_directory ()` `const`
- `bool has_root_name ()` `const`
- `bool has_root_path ()` `const`
- `bool is_relative ()` `const`
- `const string_type & native ()` `const noexcept`
- `operator string_type ()` `const`
- `template<typename _Source >`
`_Path< _Source > & operator+= (_Source const &__x)`
- `path & operator/= (const path &__p)`
- `template<class _Source >`
`_Path< _Source > & operator/= (_Source const &__source)`
- `path & operator= (const path &__p)=default`
- `template<typename _Source >`
`_Path< _Source > & operator= (_Source const &__source)`
- `path parent_path ()` `const`
- `path relative_path ()` `const`
- `path & remove_filename ()`
- `path & replace_extension (const path &__replacement=path())`
- `path & replace_filename (const path &__replacement)`
- `path root_directory ()` `const`
- `path root_name ()` `const`
- `path root_path ()` `const`
- `path & operator= (path &&__p)` `noexcept`
- `path & operator= (string_type &&__source)`
- `path & assign (string_type &&__source)`
- `path & operator+= (const path &__x)`
- `path & operator+= (const string_type &__x)`
- `path & operator+= (const value_type * __x)`
- `path & operator+= (value_type __x)`
- `path & operator+= (__basic_string_view< value_type > __x)`
- `template<typename _CharT >`
`_Path< _CharT *, _CharT * > & operator+= (_CharT __x)`
- `path & make_preferred ()`
- `void swap (path &__rhs)` `noexcept`

- `template<typename _CharT, typename _Traits = std::char_traits<_CharT>, typename _Allocator = std::allocator<_CharT>>> std::basic_string<_CharT, _Traits, _Allocator> string (const _Allocator &__a=_Allocator()) const`
- `std::string string () const`
- `std::wstring wstring () const`
- `std::string u8string () const`
- `std::u16string u16string () const`
- `std::u32string u32string () const`
- `template<typename _CharT, typename _Traits = std::char_traits<_CharT>, typename _Allocator = std::allocator<_CharT>>> std::basic_string<_CharT, _Traits, _Allocator> generic_string (const _Allocator &__a=_Allocator()) const`
- `std::string generic_string () const`
- `std::wstring generic_wstring () const`
- `std::string generic_u8string () const`
- `std::u16string generic_u16string () const`
- `std::u32string generic_u32string () const`
- `int compare (const string_type &__s) const`
- `int compare (const value_type * __s) const`
- `int compare (const __basic_string_view< value_type > __s) const`
- `path filename () const`
- `path stem () const`
- `path extension () const`
- `bool has_stem () const`
- `bool has_extension () const`
- `bool is_absolute () const`
- `iterator begin () const`
- `iterator end () const`

Static Public Attributes

- static constexpr value_type **preferred_separator**

5.696.1 Detailed Description

A filesystem path.

Definition at line 79 of file `fs_path.h`.

The documentation for this class was generated from the following file:

- [fs_path.h](#)

5.697 `std::experimental::filesystem::v1::path::iterator` Class Reference

Public Types

- using **difference_type** = `std::ptrdiff_t`
- using **iterator_category** = [std::bidirectional_iterator_tag](#)
- using **pointer** = const [path](#) *
- using **reference** = const [path](#) &
- using **value_type** = [path](#)

Public Member Functions

- **iterator** (const [iterator](#) &)=default
- [iterator](#) **operator++** (int)
- [iterator](#) **operator--** (int)
- [pointer](#) **operator->** () const
- [iterator](#) & **operator=** (const [iterator](#) &)=default

Friends

- bool **operator!=** (const [iterator](#) &__lhs, const [iterator](#) &__rhs)
- bool **operator==** (const [iterator](#) &__lhs, const [iterator](#) &__rhs)
- class **path**
- [reference](#) **operator*** () const
- [iterator](#) & **operator++** ()
- [iterator](#) & **operator--** ()

5.697.1 Detailed Description

An iterator for the components of a path.

Definition at line 709 of file `fs_path.h`.

The documentation for this class was generated from the following file:

- [fs_path.h](#)

5.698 `std::experimental::fundamentals_v1::_Has_addressof<_Tp>` Struct Template Reference

Inherits type `<_Has_addressof_mem<_Tp>, _Has_addressof_free<_Tp>>`.

5.698.1 Detailed Description

```
template<typename _Tp>
struct std::experimental::fundamentals_v1::_Has_addressof<_Tp>
```

Trait that detects the presence of an overloaded unary operator&.

Practically speaking this detects the presence of such an operator when called on a const-qualified lvalue (i.e. `declval<_Tp * const&>().operator&()`).

Definition at line 164 of file `optional`.

The documentation for this struct was generated from the following file:

- [optional](#)

5.699 `std::experimental::fundamentals_v1::_Optional_base< _Tp, _ShouldProvideDestructor >` Class Template Reference

Public Member Functions

- `constexpr _Optional_base (nullopt_t) noexcept`
- `template<typename... _Args>`
`constexpr _Optional_base (in_place_t, _Args &&...__args)`
- `template<typename _Up, typename... _Args, enable_if_t< is_constructible< _Tp, initializer_list< _Up > &, _Args &&... >::value, int > ...>`
`constexpr _Optional_base (in_place_t, initializer_list< _Up > __il, _Args &&...__args)`
- `_Optional_base (const _Optional_base &__other)`
- `_Optional_base (_Optional_base &&__other) noexcept(is_nothrow_move_constructible< _Tp >())`
- `_Optional_base & operator= (const _Optional_base &__other)`
- `_Optional_base & operator= (_Optional_base &&__other) noexcept(__and_< is_nothrow_move_constructible< _Tp >, is_nothrow_move_assignable< _Tp >>())`

Protected Member Functions

- `template<typename... _Args>`
`void _M_construct (_Args &&...__args) noexcept(is_nothrow_constructible< _Stored_type, _Args... >())`
- `void _M_destruct ()`
- `constexpr _Tp & _M_get () noexcept`
- `constexpr const _Tp & _M_get () const noexcept`
- `constexpr bool _M_is_engaged () const noexcept`
- `void _M_reset ()`

5.699.1 Detailed Description

```
template<typename _Tp, bool _ShouldProvideDestructor = !is_trivially_destructible<_Tp>::value>
class std::experimental::fundamentals_v1::_Optional_base< _Tp, _ShouldProvideDestructor >
```

Class template that holds the necessary state for [Optional values](#) and that has the responsibility for construction and the special members.

Such a separate base class template is necessary in order to conditionally enable the special members (e.g. copy/move constructors). Note that this means that [_Optional_base](#) implements the functionality for copy and move assignment, but not for converting assignment.

See also

`optional`, `_Enable_special_members`

Definition at line 204 of file `optional`.

The documentation for this class was generated from the following file:

- [optional](#)

5.700 std::experimental::fundamentals_v1::_Optional_base<_Tp, false > Class Template Reference

Public Member Functions

- constexpr **_Optional_base** ([nullopt_t](#)) noexcept
- template<typename... _Args>
constexpr **_Optional_base** ([in_place_t](#), _Args &&...__args)
- template<typename _Up, typename... _Args, enable_if_t< is_constructible< _Tp, initializer_list< _Up > &, _Args &&... >::value, int > ...>
constexpr **_Optional_base** ([in_place_t](#), [initializer_list](#)< _Up > __il, _Args &&...__args)
- **_Optional_base** (const [_Optional_base](#) &__other)
- **_Optional_base** ([_Optional_base](#) &&__other) noexcept(is_nothrow_move_constructible< _Tp >())
- [_Optional_base](#) & **operator=** (const [_Optional_base](#) &__other)
- [_Optional_base](#) & **operator=** ([_Optional_base](#) &&__other) noexcept(__and_< is_nothrow_move_constructible< _Tp >, is_nothrow_move_assignable< _Tp >>())

Protected Member Functions

- template<typename... _Args>
void **_M_construct** (_Args &&...__args) noexcept(is_nothrow_constructible< _Stored_type, _Args... >())
- void **_M_destruct** ()
- _Tp & **_M_get** () noexcept
- constexpr const _Tp & **_M_get** () const noexcept
- constexpr bool **_M_is_engaged** () const noexcept
- void **_M_reset** ()

5.700.1 Detailed Description

```
template<typename _Tp>
class std::experimental::fundamentals_v1::_Optional_base<_Tp, false >
```

Partial specialization that is exactly identical to the primary template save for not providing a destructor, to fulfill triviality requirements.

Definition at line 347 of file optional.

The documentation for this class was generated from the following file:

- [optional](#)

5.701 std::experimental::fundamentals_v1::any Class Reference

Public Member Functions

- [any](#) () noexcept
- [any](#) (const [any](#) &__other)
- [any](#) ([any](#) &&__other) noexcept
- template<typename _ValueType , typename _Tp = _Decay<_ValueType>, typename _Mgr = _Manager<_Tp>, typename enable_if< is_c<_constructible< _Tp, _ValueType && >::value, bool >::type = true>
[any](#) (_ValueType &&__value)
- template<typename _ValueType , typename _Tp = _Decay<_ValueType>, typename _Mgr = _Manager<_Tp>, typename enable_if<!is_c<_constructible< _Tp, _ValueType && >::value, bool >::type = false>
[any](#) (_ValueType &&__value)
- [~any](#) ()
- void [clear](#) () noexcept
- bool [empty](#) () const noexcept
- [any](#) & [operator=](#) (const [any](#) &__rhs)
- [any](#) & [operator=](#) ([any](#) &&__rhs) noexcept
- template<typename _ValueType >
enable_if_t<!is_same< [any](#), decay_t< _ValueType > >::value, [any](#) & > [operator=](#) (_ValueType &&__rhs)
- void [swap](#) ([any](#) &__rhs) noexcept
- const [type_info](#) & [type](#) () const noexcept

Static Public Member Functions

- template<typename _Tp >
static constexpr bool [__is_valid_cast](#) ()

Friends

- template<typename _Tp >
void * [__any_caster](#) (const [any](#) *__any)

5.701.1 Detailed Description

A type-safe container of any type.

An [any](#) object's state is either empty or it stores a contained object of CopyConstructible type.

Definition at line 89 of file [any](#).

5.701.2 Constructor & Destructor Documentation

5.701.2.1 std::experimental::fundamentals_v1::any() [inline], [noexcept]

Default constructor, creates an empty object.

Definition at line 128 of file [any](#).

5.701.2.2 `std::experimental::fundamentals_v1::any(const any &__other) [inline]`

Copy constructor, copies the state of `__other`.

Definition at line 131 of file `any`.

5.701.2.3 `std::experimental::fundamentals_v1::any(any &&__other) [inline], [noexcept]`

Move constructor, transfer the state from `__other`.

Postcondition

`__other.empty()` (this postcondition is a GNU extension)

Definition at line 148 of file `any`.

5.701.2.4 `template<typename _ValueType, typename _Tp = _Decay<_ValueType>, typename _Mgr = _Manager<_Tp>,
typename enable_if< is_constructible< _Tp, _ValueType && >::value, bool >::type = true>
std::experimental::fundamentals_v1::any(_ValueType && __value) [inline]`

Construct with a copy of `__value` as the contained object.

Definition at line 165 of file `any`.

5.701.2.5 `template<typename _ValueType, typename _Tp = _Decay<_ValueType>, typename _Mgr = _Manager<_Tp>,
typename enable_if<!is_constructible< _Tp, _ValueType && >::value, bool >::type = false>
std::experimental::fundamentals_v1::any(_ValueType && __value) [inline]`

Construct with a copy of `__value` as the contained object.

Definition at line 178 of file `any`.

5.701.2.6 `std::experimental::fundamentals_v1::any::~~any() [inline]`

Destructor, calls `clear()`

Definition at line 187 of file `any`.

5.701.3 Member Function Documentation

5.701.3.1 `void std::experimental::fundamentals_v1::any::clear() [inline], [noexcept]`

If not empty, destroy the contained object.

Definition at line 239 of file `any`.

5.701.3.2 `bool std::experimental::fundamentals_v1::any::empty () const` `[inline], [noexcept]`

Reports whether there is a contained object or not.

Definition at line 281 of file any.

5.701.3.3 `any& std::experimental::fundamentals_v1::any::operator= (const any &__rhs)` `[inline]`

Copy the state of another object.

Definition at line 192 of file any.

5.701.3.4 `any& std::experimental::fundamentals_v1::any::operator= (any &&__rhs)` `[inline], [noexcept]`

Move assignment operator.

Postcondition

`__rhs.empty()` (not guaranteed for other implementations)

Definition at line 212 of file any.

5.701.3.5 `template<typename _ValueType> enable_if_t<!is_same<any, decay_t<_ValueType>>::value, any&>`
`std::experimental::fundamentals_v1::any::operator= (_ValueType &&__rhs)` `[inline]`

Store a copy of `__rhs` as the contained object.

Definition at line 230 of file any.

5.701.3.6 `void std::experimental::fundamentals_v1::any::swap (any &__rhs)` `[inline], [noexcept]`

Exchange state with another object.

Definition at line 249 of file any.

5.701.3.7 `const type_info& std::experimental::fundamentals_v1::any::type () const` `[inline], [noexcept]`

The `typeid` of the contained object, or `typeid(void)` if empty.

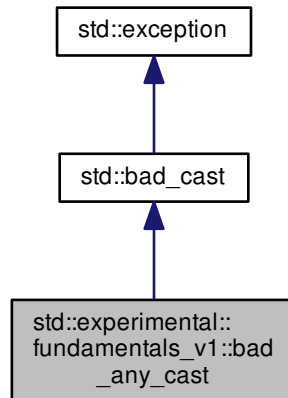
Definition at line 285 of file any.

The documentation for this class was generated from the following file:

- [any](#)

5.702 `std::experimental::fundamentals_v1::bad_any_cast` Class Reference

Inheritance diagram for `std::experimental::fundamentals_v1::bad_any_cast`:



Public Member Functions

- virtual const char * [what](#) () const noexcept

5.702.1 Detailed Description

Exception class thrown by a failed `any_cast`.

Definition at line 68 of file `any`.

5.702.2 Member Function Documentation

5.702.2.1 virtual const char* `std::experimental::fundamentals_v1::bad_any_cast::what () const` `[inline]`, `[virtual]`, `[noexcept]`

Returns a C-style character string describing the general cause of the current error.

Reimplemented from [std::bad_cast](#).

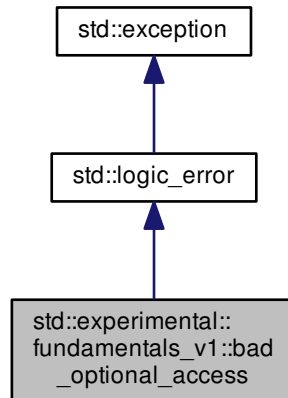
Definition at line 71 of file `any`.

The documentation for this class was generated from the following file:

- [any](#)

5.703 `std::experimental::fundamentals_v1::bad_optional_access` Class Reference

Inheritance diagram for `std::experimental::fundamentals_v1::bad_optional_access`:



Public Member Functions

- **`bad_optional_access`** (`const char *__arg`)
- virtual `const char * what () const` `_GLIBCXX_TXN_SAFE_DYN` `noexcept`

5.703.1 Detailed Description

Exception class thrown when a disengaged optional object is dereferenced.

Definition at line 117 of file `optional`.

5.703.2 Member Function Documentation

5.703.2.1 `virtual const char* std::logic_error::what () const` `[virtual]`, `[noexcept]`, `[inherited]`

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

Reimplemented in [std::future_error](#).

The documentation for this class was generated from the following file:

- [optional](#)

5.704 std::experimental::fundamentals_v1::basic_string_view<_CharT, _Traits> Class Template Reference

Public Types

- using **const_iterator** = const _CharT *
- using **const_pointer** = const _CharT *
- using **const_reference** = const _CharT &
- using **const_reverse_iterator** = [std::reverse_iterator](#)< const_iterator >
- using **difference_type** = ptrdiff_t
- using **iterator** = const_iterator
- using **pointer** = const _CharT *
- using **reference** = const _CharT &
- using **reverse_iterator** = [const_reverse_iterator](#)
- using **size_type** = size_t
- using **traits_type** = _Traits
- using **value_type** = _CharT

Public Member Functions

- constexpr **basic_string_view** (const [basic_string_view](#) &) noexcept=default
- template<typename _Allocator >
 basic_string_view (const [basic_string](#)< _CharT, _Traits, _Allocator > &__str) noexcept
- constexpr **basic_string_view** (const _CharT *__str)
- constexpr **basic_string_view** (const _CharT *__str, size_type __len)
- constexpr const _CharT & **at** (size_type __pos) const
- constexpr const _CharT & **back** () const
- constexpr const_iterator **begin** () const noexcept
- constexpr const_iterator **cbegin** () const noexcept
- constexpr const_iterator **cend** () const noexcept
- int **compare** ([basic_string_view](#) __str) const noexcept
- int **compare** (size_type __pos1, size_type __n1, [basic_string_view](#) __str) const
- int **compare** (size_type __pos1, size_type __n1, [basic_string_view](#) __str, size_type __pos2, size_type __n2) const
- int **compare** (const _CharT *__str) const noexcept
- int **compare** (size_type __pos1, size_type __n1, const _CharT *__str) const
- int **compare** (size_type __pos1, size_type __n1, const _CharT *__str, size_type __n2) const
- size_type **copy** (_CharT *__str, size_type __n, size_type __pos=0) const
- [const_reverse_iterator](#) **crbegin** () const noexcept
- [const_reverse_iterator](#) **crend** () const noexcept
- constexpr const _CharT * **data** () const noexcept
- constexpr bool **empty** () const noexcept
- constexpr const_iterator **end** () const noexcept
- size_type **find** ([basic_string_view](#) __str, size_type __pos=0) const noexcept
- size_type **find** (_CharT __c, size_type __pos=0) const noexcept
- size_type **find** (const _CharT *__str, size_type __pos, size_type __n) const noexcept
- size_type **find** (const _CharT *__str, size_type __pos=0) const noexcept
- size_type **find_first_not_of** ([basic_string_view](#) __str, size_type __pos=0) const noexcept
- size_type **find_first_not_of** (_CharT __c, size_type __pos=0) const noexcept
- size_type **find_first_not_of** (const _CharT *__str, size_type __pos, size_type __n) const
- size_type **find_first_not_of** (const _CharT *__str, size_type __pos=0) const noexcept

- `size_type find_first_of (basic_string_view __str, size_type __pos=0) const noexcept`
- `size_type find_first_of (_CharT __c, size_type __pos=0) const noexcept`
- `size_type find_first_of (const _CharT * __str, size_type __pos, size_type __n) const`
- `size_type find_first_of (const _CharT * __str, size_type __pos=0) const noexcept`
- `size_type find_last_not_of (basic_string_view __str, size_type __pos=npos) const noexcept`
- `size_type find_last_not_of (_CharT __c, size_type __pos=npos) const noexcept`
- `size_type find_last_not_of (const _CharT * __str, size_type __pos, size_type __n) const`
- `size_type find_last_not_of (const _CharT * __str, size_type __pos=npos) const noexcept`
- `size_type find_last_of (basic_string_view __str, size_type __pos=npos) const noexcept`
- `size_type find_last_of (_CharT __c, size_type __pos=npos) const noexcept`
- `size_type find_last_of (const _CharT * __str, size_type __pos, size_type __n) const`
- `size_type find_last_of (const _CharT * __str, size_type __pos=npos) const noexcept`
- `constexpr const _CharT & front () const`
- `constexpr size_type length () const noexcept`
- `constexpr size_type max_size () const noexcept`
- `template<typename _Allocator >
operator basic_string< _CharT, _Traits, _Allocator > () const`
- `basic_string_view & operator= (const basic_string_view &) noexcept=default`
- `constexpr const _CharT & operator[] (size_type __pos) const`
- `const_reverse_iterator rbegin () const noexcept`
- `void remove_prefix (size_type __n)`
- `void remove_suffix (size_type __n)`
- `const_reverse_iterator rend () const noexcept`
- `size_type rfind (basic_string_view __str, size_type __pos=npos) const noexcept`
- `size_type rfind (_CharT __c, size_type __pos=npos) const noexcept`
- `size_type rfind (const _CharT * __str, size_type __pos, size_type __n) const noexcept`
- `size_type rfind (const _CharT * __str, size_type __pos=npos) const noexcept`
- `constexpr size_type size () const noexcept`
- `constexpr basic_string_view substr (size_type __pos, size_type __n=npos) const`
- `void swap (basic_string_view & __sv) noexcept`
- `template<typename _Allocator = std::allocator<_CharT>>
basic_string< _CharT, _Traits, _Allocator > to_string (const _Allocator & __alloc=_Allocator()) const`

Static Public Attributes

- `static constexpr size_type npos`

5.704.1 Detailed Description

```
template<typename _CharT, typename _Traits = std::char_traits<_CharT>>
class std::experimental::fundamentals_v1::basic_string_view< _CharT, _Traits >
```

A non-owning reference to a string.

Template Parameters

<code>_CharT</code>	Type of character
<code>_Traits</code>	Traits for character type, defaults to <code>char_traits<_CharT></code> .

A basic_string_view looks like this:

```
1 _CharT*      _M_str
2 size_t      _M_len
```

Definition at line 76 of file string_view.

The documentation for this class was generated from the following files:

- [string_view](#)
- [string_view.tcc](#)

5.705 std::experimental::fundamentals_v1::in_place_t Struct Reference

5.705.1 Detailed Description

Tag type for in-place construction.

Definition at line 87 of file optional.

The documentation for this struct was generated from the following file:

- [optional](#)

5.706 std::experimental::fundamentals_v1::nullopt_t Struct Reference

Public Types

- enum **_Construct** { **_Token** }

Public Member Functions

- constexpr **nullopt_t** (_Construct)

5.706.1 Detailed Description

Tag type to disengage optional objects.

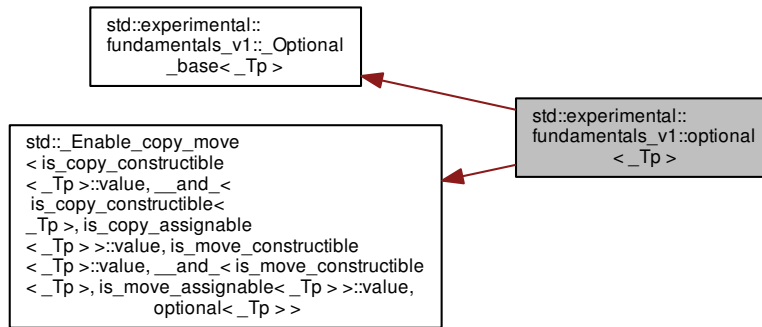
Definition at line 94 of file optional.

The documentation for this struct was generated from the following file:

- [optional](#)

5.707 std::experimental::fundamentals_v1::optional<_Tp> Class Template Reference

Inheritance diagram for std::experimental::fundamentals_v1::optional<_Tp>:



Public Types

- using **value_type** = `_Tp`

Public Member Functions

- `template<typename _Up = _Tp, enable_if_t<__and<__not<is_same<optional<_Tp>, decay_t<_Up>>>, is_constructible<_Tp, _Up &&>, is_convertible<_Up &&, _Tp>>::value, bool> = true>`
`constexpr optional (&&_Up &&__t)`
- `template<typename _Up = _Tp, enable_if_t<__and<__not<is_same<optional<_Tp>, decay_t<_Up>>>, is_constructible<_Tp, _Up &&>, __not<is_convertible<_Up &&, _Tp>>::value, bool> = false>`
`constexpr optional (&&_Up &&__t)`
- `template<typename _Up, enable_if_t<__and<__not<is_same<_Tp, _Up>>, is_constructible<_Tp, const _Up &>, is_convertible<const _Up &, _Tp>, __not<__converts_from_optional<_Tp, _Up>>::value, bool> = true>`
`constexpr optional (const optional<_Up> &__t)`
- `template<typename _Up, enable_if_t<__and<__not<is_same<_Tp, _Up>>, is_constructible<_Tp, const _Up &>, __not<is_convertible<const _Up &, _Tp>>, __not<__converts_from_optional<_Tp, _Up>>::value, bool> = false>`
`constexpr optional (const optional<_Up> &__t)`
- `template<typename _Up, enable_if_t<__and<__not<is_same<_Tp, _Up>>, is_constructible<_Tp, _Up &&>, is_convertible<_Up &&, _Tp>, __not<__converts_from_optional<_Tp, _Up>>::value, bool> = true>`
`constexpr optional (optional<_Up> &&__t)`
- `template<typename _Up, enable_if_t<__and<__not<is_same<_Tp, _Up>>, is_constructible<_Tp, _Up &&>, __not<is_convertible<_Up &&, _Tp>>, __not<__converts_from_optional<_Tp, _Up>>::value, bool> = false>`
`constexpr optional (optional<_Up> &&__t)`
- `template<typename... _Args>`
`enable_if_t<is_constructible<_Tp, _Args &&...>::value> emplace (_Args &&... __args)`
- `template<typename _Up, typename... _Args>`
`enable_if_t<is_constructible<_Tp, initializer_list<_Up> &, _Args &&...>::value> emplace (initializer_list<_Up> &__il, _Args &&... __args)`

- constexpr **operator bool** () const noexcept
- constexpr const _Tp & **operator*** () const &
- constexpr _Tp & **operator*** ()&
- constexpr _Tp && **operator*** ()&&
- constexpr const _Tp && **operator*** () const &&
- constexpr const _Tp * **operator->** () const
- _Tp * **operator->** ()
- **optional** & **operator=** (nullopt_t) noexcept
- template<typename _Up = _Tp>
enable_if_t<__and<__not<is_same<optional<_Tp>, decay_t<_Up>>>, is_constructible<_Tp, _Up>, __not<__and<is_scalar<_Tp>, is_same<_Tp, decay_t<_Up>>>>, is_assignable<_Tp &, _Up>>::value, optional &> **operator=** (_Up &&__u)
- template<typename _Up >
enable_if_t<__and<__not<is_same<_Tp, _Up>>, is_constructible<_Tp, const _Up &>, is_assignable<_Tp &, _Up>, __not<__converts_from_optional<_Tp, _Up>>, __not<__assigns_from_optional<_Tp, _Up>>::value, optional &> **operator=** (const optional<_Up> &__u)
- template<typename _Up >
enable_if_t<__and<__not<is_same<_Tp, _Up>>, is_constructible<_Tp, _Up>, is_assignable<_Tp &, _Up>, __not<__converts_from_optional<_Tp, _Up>>, __not<__assigns_from_optional<_Tp, _Up>>::value, optional &> **operator=** (optional<_Up> &&__u)
- void **swap** (optional &__other) noexcept(is_nothrow_move_constructible<_Tp>() &&noexcept(swap(declval<_Tp> &>(), declval<_Tp> &>())))
- constexpr const _Tp & **value** () const &
- constexpr _Tp & **value** ()&
- constexpr _Tp && **value** ()&&
- constexpr const _Tp && **value** () const &&
- template<typename _Up >
constexpr _Tp **value_or** (_Up &&__u) const &
- template<typename _Up >
_Tp **value_or** (_Up &&__u)&&

Private Member Functions

- void **_M_construct** (_Args &&...__args) noexcept(is_nothrow_constructible<_Stored_type, _Args...>())
- void **_M_destruct** ()
- constexpr _Tp & **_M_get** () noexcept
- constexpr const _Tp & **_M_get** () const noexcept
- constexpr bool **_M_is_engaged** () const noexcept
- void **_M_reset** ()

Private Attributes

- _Empty_byte **_M_empty**
- _Stored_type **_M_payload**

5.707.1 Detailed Description

```
template<typename _Tp>
class std::experimental::fundamentals_v1::optional< _Tp >
```

Class template for optional values.

Definition at line 83 of file optional.

The documentation for this class was generated from the following file:

- [optional](#)

5.708 std::experimental::fundamentals_v2::_Not_fn< _Fn > Class Template Reference

Public Member Functions

- template<typename _Fn2 >
 _Not_fn (_Fn2 && __fn, int)
- **_Not_fn** (const [_Not_fn](#) & __fn)=default
- **_Not_fn** ([_Not_fn](#) && __fn)=default
- template<typename... _Args>
 auto **operator()** (_Args &&... __args) noexcept(noexcept(!_M_fn([std::forward](#)< _Args >(__args)...))) ->
 decltype(!_M_fn([std::forward](#)< _Args >(__args)...))
- template<typename... _Args>
 auto **operator()** (_Args &&... __args) const noexcept(noexcept(!_M_fn([std::forward](#)< _Args >(__args)...))) ->
 decltype(!_M_fn([std::forward](#)< _Args >(__args)...))
- template<typename... _Args>
 auto **operator()** (_Args &&... __args) volatile noexcept(noexcept(!_M_fn([std::forward](#)< _Args >(__args)...))) ->
 decltype(!_M_fn([std::forward](#)< _Args >(__args)...))
- template<typename... _Args>
 auto **operator()** (_Args &&... __args) const volatile noexcept(noexcept(!_M_fn([std::forward](#)< _Args >(__args)...))) ->
 decltype(!_M_fn([std::forward](#)< _Args >(__args)...))
- [_Not_fn](#) & **operator=** (const [_Not_fn](#) & __fn)=default
- [_Not_fn](#) & **operator=** ([_Not_fn](#) && __fn)=default

5.708.1 Detailed Description

```
template<typename _Fn>
class std::experimental::fundamentals_v2::_Not_fn< _Fn >
```

Generalized negator.

Definition at line 383 of file experimental/functional.

The documentation for this class was generated from the following file:

- [experimental/functional](#)

5.709 `std::experimental::fundamentals_v2::ostream_joiner<_DelimT, _CharT, _Traits >` Class Template Reference

Public Types

- typedef `_CharT` **char_type**
- typedef void **difference_type**
- typedef [output_iterator_tag](#) **iterator_category**
- typedef [basic_ostream](#)< `_CharT`, `_Traits` > **ostream_type**
- typedef void **pointer**
- typedef void **reference**
- typedef `_Traits` **traits_type**
- typedef void **value_type**

Public Member Functions

- **ostream_joiner** ([ostream_type](#) &__os, const `_DelimT` &__delimiter) noexcept(is_nothrow_copy_constructible_v< `_DelimT` >)
- **ostream_joiner** ([ostream_type](#) &__os, `_DelimT` &&__delimiter) noexcept(is_nothrow_move_constructible_v< `_DelimT` >)
- [ostream_joiner](#) & **operator*** () noexcept
- [ostream_joiner](#) & **operator++** () noexcept
- [ostream_joiner](#) & **operator++** (int) noexcept
- template<typename `_Tp` >
[ostream_joiner](#) & **operator=** (const `_Tp` &__value)

5.709.1 Detailed Description

```
template<typename _DelimT, typename _CharT = char, typename _Traits = char_traits<_CharT>>
class std::experimental::fundamentals_v2::ostream_joiner<_DelimT, _CharT, _Traits >
```

Output iterator that inserts a delimiter between elements.

Definition at line 60 of file `experimental/iterator`.

The documentation for this class was generated from the following file:

- [experimental/iterator](#)

5.710 `std::experimental::fundamentals_v2::owner_less< shared_ptr<_Tp> >` Struct Template Reference

Inherits `std::_Sp_owner_less<_Tp, _Tp1 >`.

Public Types

- typedef `_Tp` [first_argument_type](#)
- typedef `bool` [result_type](#)
- typedef `_Tp` [second_argument_type](#)

Public Member Functions

- `bool operator()` (`const _Tp &__lhs, const _Tp &__rhs`) `const`
- `bool operator()` (`const _Tp &__lhs, const _Tp1 &__rhs`) `const`
- `bool operator()` (`const _Tp1 &__lhs, const _Tp &__rhs`) `const`

5.710.1 Detailed Description

```
template<typename _Tp>
struct std::experimental::fundamentals_v2::owner_less< shared_ptr< _Tp > >
```

Partial specialization of `owner_less` for `shared_ptr`.

Definition at line 1151 of file `experimental/bits/shared_ptr.h`.

5.710.2 Member Typedef Documentation

5.710.2.1 `typedef _Tp std::binary_function< _Tp, _Tp, bool >::first_argument_type` `[inherited]`

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.710.2.2 `typedef bool std::binary_function< _Tp, _Tp, bool >::result_type` `[inherited]`

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.710.2.3 `typedef _Tp std::binary_function< _Tp, _Tp, bool >::second_argument_type` `[inherited]`

`second_argument_type` is the type of the second argument

Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [experimental/bits/shared_ptr.h](#)

5.711 std::experimental::fundamentals_v2::owner_less< weak_ptr< _Tp > > Struct Template Reference

Inherits std::_Sp_owner_less< _Tp, _Tp1 >.

Public Types

- typedef _Tp [first_argument_type](#)
- typedef bool [result_type](#)
- typedef _Tp [second_argument_type](#)

Public Member Functions

- bool **operator()** (const _Tp &__lhs, const _Tp &__rhs) const
- bool **operator()** (const _Tp &__lhs, const _Tp1 &__rhs) const
- bool **operator()** (const _Tp1 &__lhs, const _Tp &__rhs) const

5.711.1 Detailed Description

```
template<typename _Tp>
struct std::experimental::fundamentals_v2::owner_less< weak_ptr< _Tp > >
```

Partial specialization of owner_less for weak_ptr.

Definition at line 1157 of file experimental/bits/shared_ptr.h.

5.711.2 Member Typedef Documentation

5.711.2.1 typedef _Tp std::binary_function< _Tp, _Tp, bool >::first_argument_type [inherited]

first_argument_type is the type of the first argument

Definition at line 121 of file stl_function.h.

5.711.2.2 typedef bool std::binary_function< _Tp, _Tp, bool >::result_type [inherited]

result_type is the return type

Definition at line 127 of file stl_function.h.

5.711.2.3 typedef _Tp std::binary_function< _Tp, _Tp, bool >::second_argument_type [inherited]

second_argument_type is the type of the second argument

Definition at line 124 of file stl_function.h.

The documentation for this struct was generated from the following file:

- [experimental/bits/shared_ptr.h](#)

5.712 `std::experimental::fundamentals_v2::propagate_const<_Tp>` Class Template Reference

Public Types

- typedef `remove_reference_t< decltype(*std::declval<_Tp &>)>` **element_type**

Public Member Functions

- **propagate_const** (const [propagate_const](#) &__p)=delete
- constexpr **propagate_const** ([propagate_const](#) &&__p)=default
- template<typename _Up , typename enable_if< __and_< is_constructible< _Tp, _Up && >, is_convertible< _Up &&, _Tp >>::value, bool >::type = true>
constexpr **propagate_const** ([propagate_const](#)< _Up > &&__pu)
- template<typename _Up , typename enable_if< __and_< is_constructible< _Tp, _Up && >, __not_< is_convertible< _Up &&, _Tp >>::value, bool >::type = false>
constexpr **propagate_const** ([propagate_const](#)< _Up > &&__pu)
- template<typename _Up , typename enable_if< __and_< is_constructible< _Tp, _Up && >, is_convertible< _Up &&, _Tp >, __not_< __is_propagate_const< typename decay< _Up >::type >> >::value, bool >::type = true>
constexpr **propagate_const** (_Up &&__u)
- template<typename _Up , typename enable_if< __and_< is_constructible< _Tp, _Up && >, __not_< is_convertible< _Up &&, _Tp >>, __not_< __is_propagate_const< typename decay< _Up >::type >> >::value, bool >::type = false>
constexpr **propagate_const** (_Up &&__u)
- constexpr const element_type * **get** () const
- constexpr element_type * **get** ()
- constexpr **operator bool** () const
- template<typename _Up = _Tp, typename enable_if< __or_< is_pointer< _Up >, is_convertible< _Up, const element_type * >>::value, bool >::type = true>
constexpr **operator const element_type *** () const
- template<typename _Up = _Tp, typename enable_if< __or_< is_pointer< _Up >, is_convertible< _Up, const element_type * >>::value, bool >::type = true>
constexpr **operator element_type *** ()
- constexpr const element_type & **operator*** () const
- constexpr element_type & **operator*** ()
- constexpr const element_type * **operator->** () const
- constexpr element_type * **operator->** ()
- [propagate_const](#) & **operator=** (const [propagate_const](#) &__p)=delete
- constexpr [propagate_const](#) & **operator=** ([propagate_const](#) &&__p)=default
- template<typename _Up , typename = typename enable_if<is_convertible<_Up&&, _Tp>::value>::type>
constexpr [propagate_const](#) & **operator=** ([propagate_const](#)< _Up > &&__pu)
- template<typename _Up , typename = typename enable_if<__and_<is_convertible<_Up&&, _Tp>, __not_<__is_propagate_const<typename decay<_Up>::type>> >::value>::type>
constexpr [propagate_const](#) & **operator=** (_Up &&__u)
- constexpr void **swap** ([propagate_const](#) &__pt) noexcept(__is_nothrow_swappable< _Tp >::value)

Friends

- template<typename _Up >
constexpr const _Up & **get_underlying** (const [propagate_const](#)< _Up > &__pt) noexcept
- template<typename _Up >
constexpr _Up & **get_underlying** ([propagate_const](#)< _Up > &__pt) noexcept

5.712.1 Detailed Description

```
template<typename _Tp>
class std::experimental::fundamentals_v2::propagate_const<_Tp>
```

Const-propagating wrapper.

Definition at line 63 of file `propagate_const`.

The documentation for this class was generated from the following file:

- [propagate_const](#)

5.713 `std::exponential_distribution<_RealType>` Class Template Reference

Classes

- struct [param_type](#)

Public Types

- typedef `_RealType` [result_type](#)

Public Member Functions

- [exponential_distribution](#) (const [result_type](#) &__lambda=[result_type](#)(1))
- [exponential_distribution](#) (const [param_type](#) &__p)
- `template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >`
void [__generate](#) (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)
- `template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >`
void [__generate](#) (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- `template<typename _UniformRandomNumberGenerator >`
void [__generate](#) ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- `_RealType` [lambda](#) () const
- [result_type](#) [max](#) () const
- [result_type](#) [min](#) () const
- `template<typename _UniformRandomNumberGenerator >`
[result_type](#) [operator\(\)](#) (_UniformRandomNumberGenerator &__urng)
- `template<typename _UniformRandomNumberGenerator >`
[result_type](#) [operator\(\)](#) (_UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [param_type](#) [param](#) () const
- void [param](#) (const [param_type](#) &__param)
- void [reset](#) ()

Friends

- bool `operator==` (const `exponential_distribution` &__d1, const `exponential_distribution` &__d2)

5.713.1 Detailed Description

```
template<typename _RealType = double>
class std::exponential_distribution< _RealType >
```

An exponential continuous distribution for random numbers.

The formula for the exponential probability density function is $p(x|\lambda) = \lambda e^{-\lambda x}$.

Table 1978 Distribution Statistics

Mean	$\frac{1}{\lambda}$
Median	$\frac{\ln 2}{\lambda}$
Mode	<i>zero</i>
Range	$[0, \infty]$
Standard Deviation	$\frac{1}{\lambda}$

Definition at line 4481 of file random.h.

5.713.2 Member Typedef Documentation

5.713.2.1 `template<typename _RealType = double> typedef _RealType std::exponential_distribution< _RealType >::result_type`

The type of the range of the distribution.

Definition at line 4484 of file random.h.

5.713.3 Constructor & Destructor Documentation

5.713.3.1 `template<typename _RealType = double> std::exponential_distribution< _RealType >::exponential_distribution (const result_type & __lambda = result_type(1)) [inline], [explicit]`

Constructs an exponential distribution with inverse scale parameter λ .

Definition at line 4519 of file random.h.

5.713.4 Member Function Documentation

5.713.4.1 `template<typename _RealType = double> _RealType std::exponential_distribution<_RealType>::lambda ()`
`const [inline]`

Returns the inverse scale parameter of the distribution.

Definition at line 4540 of file `random.h`.

5.713.4.2 `template<typename _RealType = double> result_type std::exponential_distribution<_RealType>::max ()`
`const [inline]`

Returns the least upper bound value of the distribution.

Definition at line 4569 of file `random.h`.

References `std::numeric_limits<_Tp>::max()`.

5.713.4.3 `template<typename _RealType = double> result_type std::exponential_distribution<_RealType>::min ()`
`const [inline]`

Returns the greatest lower bound value of the distribution.

Definition at line 4562 of file `random.h`.

5.713.4.4 `template<typename _RealType = double> template<typename _UniformRandomNumberGenerator> result_type`
`std::exponential_distribution<_RealType>::operator() (_UniformRandomNumberGenerator & __urng)`
`[inline]`

Generating functions.

Definition at line 4577 of file `random.h`.

References `std::log()`.

5.713.4.5 `template<typename _RealType = double> param_type std::exponential_distribution<_RealType>::param ()`
`const [inline]`

Returns the parameter set of the distribution.

Definition at line 4547 of file `random.h`.

Referenced by `std::operator>>()`.

5.713.4.6 `template<typename _RealType = double> void std::exponential_distribution<_RealType>::param (const`
`param_type & __param) [inline]`

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

Definition at line 4555 of file random.h.

5.713.4.7 `template<typename _RealType = double> void std::exponential_distribution< _RealType >::reset ()`
`[inline]`

Resets the distribution state.

Has no effect on exponential distributions.

Definition at line 4534 of file random.h.

5.713.5 Friends And Related Function Documentation

5.713.5.1 `template<typename _RealType = double> bool operator==(const exponential_distribution< _RealType > &__d1,`
`const exponential_distribution< _RealType > &__d2) [friend]`

Return true if two exponential distributions have the same parameters.

Definition at line 4617 of file random.h.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.714 `std::exponential_distribution< _RealType >::param_type` Struct Reference

Public Types

- typedef `exponential_distribution< _RealType >` **distribution_type**

Public Member Functions

- **param_type** (`_RealType __lambda=_RealType(1)`)
- `_RealType lambda () const`

Friends

- bool **operator==** (const `param_type` &__p1, const `param_type` &__p2)

5.714.1 Detailed Description

```
template<typename _RealType = double>
struct std::exponential_distribution<_RealType>::param_type
```

Parameter type.

Definition at line 4490 of file random.h.

The documentation for this struct was generated from the following file:

- [random.h](#)

5.715 std::extreme_value_distribution<_RealType> Class Template Reference

Classes

- struct [param_type](#)

Public Types

- typedef _RealType [result_type](#)

Public Member Functions

- **extreme_value_distribution** (_RealType __a=_RealType(0), _RealType __b=_RealType(1))
- **extreme_value_distribution** (const [param_type](#) &__p)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- template<typename _UniformRandomNumberGenerator >
void **generate** ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- _RealType **a** () const
- _RealType **b** () const
- [result_type](#) **max** () const
- [result_type](#) **min** () const
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [param_type](#) **param** () const
- void **param** (const [param_type](#) &__param)
- void **reset** ()

Friends

- bool `operator==` (const `extreme_value_distribution` &__d1, const `extreme_value_distribution` &__d2)

5.715.1 Detailed Description

```
template<typename _RealType = double>
class std::extreme_value_distribution< _RealType >
```

A `extreme_value_distribution` random number distribution.

The formula for the normal probability mass function is

$$p(x|a, b) = \frac{1}{b} \exp\left(\frac{a-x}{b} - \exp\left(\frac{a-x}{b}\right)\right)$$

Definition at line 4886 of file `random.h`.

5.715.2 Member Typedef Documentation

5.715.2.1 `template<typename _RealType = double> typedef _RealType std::extreme_value_distribution< _RealType >::result_type`

The type of the range of the distribution.

Definition at line 4889 of file `random.h`.

5.715.3 Member Function Documentation

5.715.3.1 `template<typename _RealType = double> _RealType std::extreme_value_distribution< _RealType >::a () const`
[`inline`]

Return the a parameter of the distribution.

Definition at line 4944 of file `random.h`.

5.715.3.2 `template<typename _RealType = double> _RealType std::extreme_value_distribution< _RealType >::b () const`
[`inline`]

Return the b parameter of the distribution.

Definition at line 4951 of file `random.h`.

```
5.715.3.3 template<typename _RealType = double> result_type std::extreme_value_distribution<_RealType>::max ( )
const [inline]
```

Returns the least upper bound value of the distribution.

Definition at line 4980 of file random.h.

References std::numeric_limits<_Tp>::max().

```
5.715.3.4 template<typename _RealType = double> result_type std::extreme_value_distribution<_RealType>::min ( )
const [inline]
```

Returns the greatest lower bound value of the distribution.

Definition at line 4973 of file random.h.

References std::numeric_limits<_Tp>::lowest().

```
5.715.3.5 template<typename _RealType = double> template<typename _UniformRandomNumberGenerator> result_type
std::extreme_value_distribution<_RealType>::operator() ( _UniformRandomNumberGenerator & __urng )
[inline]
```

Generating functions.

Definition at line 4988 of file random.h.

Referenced by std::operator>>().

```
5.715.3.6 template<typename _RealType = double> param_type std::extreme_value_distribution<_RealType>::param (
) const [inline]
```

Returns the parameter set of the distribution.

Definition at line 4958 of file random.h.

Referenced by std::operator>>().

```
5.715.3.7 template<typename _RealType = double> void std::extreme_value_distribution<_RealType>::param ( const
param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

Definition at line 4966 of file random.h.

5.715.3.8 `template<typename _RealType = double> void std::extreme_value_distribution< _RealType >::reset ()`
`[inline]`

Resets the distribution state.

Definition at line 4937 of file random.h.

5.715.4 Friends And Related Function Documentation

5.715.4.1 `template<typename _RealType = double> bool operator==(const extreme_value_distribution< _RealType > & __d1, const extreme_value_distribution< _RealType > & __d2)` `[friend]`

Return true if two extreme value distributions have the same parameters.

Definition at line 5023 of file random.h.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.716 std::extreme_value_distribution< _RealType >::param_type Struct Reference

Public Types

- typedef [extreme_value_distribution< _RealType >](#) **distribution_type**

Public Member Functions

- **param_type** ([_RealType](#) __a=[_RealType](#)(0), [_RealType](#) __b=[_RealType](#)(1))
- [_RealType](#) **a** () const
- [_RealType **b** \(\) const](#)

Friends

- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)

5.716.1 Detailed Description

```
template<typename _RealType = double>
struct std::extreme_value_distribution< _RealType >::param_type
```

Parameter type.

Definition at line 4895 of file random.h.

The documentation for this struct was generated from the following file:

- [random.h](#)

5.717 std::fisher_f_distribution<_RealType> Class Template Reference

Classes

- struct [param_type](#)

Public Types

- typedef _RealType [result_type](#)

Public Member Functions

- **fisher_f_distribution** (_RealType __m=_RealType(1), _RealType __n=_RealType(1))
- **fisher_f_distribution** (const [param_type](#) &__p)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- template<typename _UniformRandomNumberGenerator >
void **generate** ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
void **generate** ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- _RealType **m** () const
- [result_type](#) **max** () const
- [result_type](#) **min** () const
- _RealType **n** () const
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [param_type](#) **param** () const
- void **param** (const [param_type](#) &__param)
- void **reset** ()

Friends

- template<typename _RealType1, typename _CharT, typename _Traits >
[std::basic_ostream](#)< _CharT, _Traits > & **operator<<** ([std::basic_ostream](#)< _CharT, _Traits > &__os, const [std::fisher_f_distribution](#)< _RealType1 > &__x)
- bool **operator==** (const [fisher_f_distribution](#) &__d1, const [fisher_f_distribution](#) &__d2)
- template<typename _RealType1, typename _CharT, typename _Traits >
[std::basic_istream](#)< _CharT, _Traits > & **operator>>** ([std::basic_istream](#)< _CharT, _Traits > &__is, [std::fisher_f_distribution](#)< _RealType1 > &__x)

5.717.1 Detailed Description

```
template<typename _RealType = double>
class std::fisher_f_distribution< _RealType >
```

A fisher_f_distribution random number distribution.

The formula for the normal probability mass function is

$$p(x|m, n) = \frac{\Gamma((m+n)/2)}{\Gamma(m/2)\Gamma(n/2)} \left(\frac{m}{n}\right)^{m/2} x^{(m/2)-1} \left(1 + \frac{mx}{n}\right)^{-(m+n)/2}$$

Definition at line 2965 of file random.h.

5.717.2 Member Typedef Documentation

```
5.717.2.1 template<typename _RealType = double> typedef _RealType std::fisher_f_distribution< _RealType
>::result_type
```

The type of the range of the distribution.

Definition at line 2968 of file random.h.

5.717.3 Member Function Documentation

```
5.717.3.1 template<typename _RealType = double> result_type std::fisher_f_distribution< _RealType >::max ( ) const
[inline]
```

Returns the least upper bound value of the distribution.

Definition at line 3059 of file random.h.

References std::numeric_limits< _Tp >::max().

```
5.717.3.2 template<typename _RealType = double> result_type std::fisher_f_distribution< _RealType >::min ( ) const
[inline]
```

Returns the greatest lower bound value of the distribution.

Definition at line 3052 of file random.h.

```
5.717.3.3 template<typename _RealType = double> template<typename _UniformRandomNumberGenerator > result_type
std::fisher_f_distribution< _RealType >::operator() ( _UniformRandomNumberGenerator & __urng ) [inline]
```

Generating functions.

Definition at line 3067 of file random.h.

```
5.717.3.4 template<typename _RealType = double> param_type std::fisher_f_distribution< _RealType >::param ( ) const
[inline]
```

Returns the parameter set of the distribution.

Definition at line 3037 of file random.h.

Referenced by std::operator>>().

```
5.717.3.5 template<typename _RealType = double> void std::fisher_f_distribution< _RealType >::param ( const
param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

Definition at line 3045 of file random.h.

5.717.3.6 `template<typename _RealType = double> void std::fisher_f_distribution<_RealType>::reset() [inline]`

Resets the distribution state.

Definition at line 3016 of file random.h.

5.717.4 Friends And Related Function Documentation

5.717.4.1 `template<typename _RealType = double> template<typename _RealType1 , typename _CharT , typename _Traits > std::basic_ostream<_CharT, _Traits>& operator<< (std::basic_ostream<_CharT, _Traits> & __os, const std::fisher_f_distribution<_RealType1> & __x) [friend]`

Inserts a fisher_f_distribution random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A fisher_f_distribution random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

5.717.4.2 `template<typename _RealType = double> bool operator==(const fisher_f_distribution<_RealType> & __d1, const fisher_f_distribution<_RealType> & __d2) [friend]`

Return true if two Fisher f distributions have the same parameters and the sequences that would be generated are equal.

Definition at line 3115 of file random.h.

5.717.4.3 `template<typename _RealType = double> template<typename _RealType1 , typename _CharT , typename _Traits > std::basic_istream<_CharT, _Traits>& operator>> (std::basic_istream<_CharT, _Traits> & __is, std::fisher_f_distribution<_RealType1> & __x) [friend]`

Extracts a fisher_f_distribution random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>fisher_f_distribution</code> random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.718 `std::fisher_f_distribution<_RealType>::param_type` Struct Reference

Public Types

- typedef `fisher_f_distribution<_RealType>` **distribution_type**

Public Member Functions

- **param_type** (`_RealType __m=_RealType(1), _RealType __n=_RealType(1)`)
- `_RealType m` () const
- `_RealType n` () const

Friends

- bool **operator==** (const `param_type` &__p1, const `param_type` &__p2)

5.718.1 Detailed Description

```
template<typename _RealType = double>
struct std::fisher_f_distribution<_RealType>::param_type
```

Parameter type.

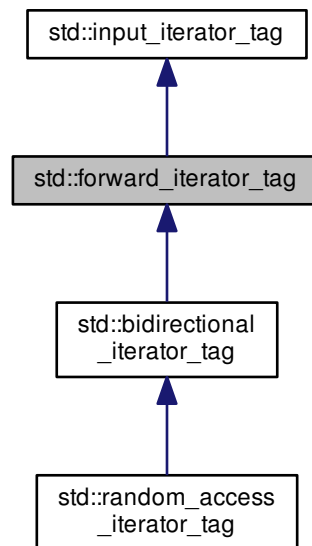
Definition at line 2974 of file `random.h`.

The documentation for this struct was generated from the following file:

- [random.h](#)

5.719 std::forward_iterator_tag Struct Reference

Inheritance diagram for std::forward_iterator_tag:



5.719.1 Detailed Description

Forward iterators support a superset of input iterator operations.

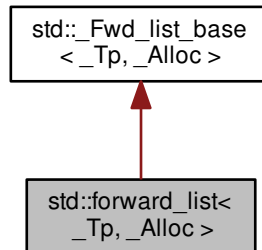
Definition at line 95 of file `stl_iterator_base_types.h`.

The documentation for this struct was generated from the following file:

- [stl_iterator_base_types.h](#)

5.720 `std::forward_list<_Tp, _Alloc>` Class Template Reference

Inheritance diagram for `std::forward_list<_Tp, _Alloc>`:



Public Types

- typedef `_Alloc` **allocator_type**
- typedef `_Fwd_list_const_iterator<_Tp>` **const_iterator**
- typedef `_Alloc_traits::const_pointer` **const_pointer**
- typedef `const value_type &` **const_reference**
- typedef `std::ptrdiff_t` **difference_type**
- typedef `_Fwd_list_iterator<_Tp>` **iterator**
- typedef `_Alloc_traits::pointer` **pointer**
- typedef `value_type &` **reference**
- typedef `std::size_t` **size_type**
- typedef `_Tp` **value_type**

Public Member Functions

- `forward_list()` noexcept(is_nothrow_default_constructible<_Node_alloc_type>::value)
- `forward_list(const _Alloc &__al)` noexcept
- `forward_list(const forward_list &__list, const _Alloc &__al)`
- `forward_list(forward_list &&__list, const _Alloc &__al)` noexcept(_Node_alloc_traits::_S_always_equal())
- `forward_list(size_type __n, const _Alloc &__al= _Alloc())`
- `forward_list(size_type __n, const _Tp &__value, const _Alloc &__al= _Alloc())`
- template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>>
 `forward_list(_InputIterator __first, _InputIterator __last, const _Alloc &__al= _Alloc())`
- `forward_list(const forward_list &__list)`
- `forward_list(forward_list &&__list)` noexcept
- `forward_list(std::initializer_list<_Tp> __il, const _Alloc &__al= _Alloc())`
- `~forward_list()` noexcept
- template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>>
 void `assign(_InputIterator __first, _InputIterator __last)`

- void `assign` (size_type __n, const _Tp &__val)
- void `assign` (std::initializer_list<_Tp> __il)
- iterator `before_begin` () noexcept
- const_iterator `before_begin` () const noexcept
- iterator `begin` () noexcept
- const_iterator `begin` () const noexcept
- const_iterator `cbegin` () const noexcept
- const_iterator `cbegin` () const noexcept
- const_iterator `cend` () const noexcept
- void `clear` () noexcept
- template<typename... _Args>
iterator `emplace_after` (const_iterator __pos, _Args &&... __args)
- template<typename... _Args>
void `emplace_front` (_Args &&... __args)
- bool `empty` () const noexcept
- iterator `end` () noexcept
- const_iterator `end` () const noexcept
- iterator `erase_after` (const_iterator __pos)
- iterator `erase_after` (const_iterator __pos, const_iterator __last)
- reference `front` ()
- const_reference `front` () const
- allocator_type `get_allocator` () const noexcept
- iterator `insert_after` (const_iterator __pos, const _Tp &__val)
- iterator `insert_after` (const_iterator __pos, _Tp &&__val)
- iterator `insert_after` (const_iterator __pos, size_type __n, const _Tp &__val)
- template<typename _InputIterator, typename = std::RequireInputIter<_InputIterator>>>
iterator `insert_after` (const_iterator __pos, _InputIterator __first, _InputIterator __last)
- iterator `insert_after` (const_iterator __pos, std::initializer_list<_Tp> __il)
- size_type `max_size` () const noexcept
- void `merge` (forward_list &&__list)
- void `merge` (forward_list &__list)
- template<typename _Comp>
void `merge` (forward_list &&__list, _Comp __comp)
- template<typename _Comp>
void `merge` (forward_list &__list, _Comp __comp)
- forward_list & `operator=` (const forward_list &__list)
- forward_list & `operator=` (forward_list &&__list) noexcept(_Node_alloc_traits::_S_nothrow_move())
- forward_list & `operator=` (std::initializer_list<_Tp> __il)
- void `pop_front` ()
- void `push_front` (const _Tp &__val)
- void `push_front` (_Tp &&__val)
- void `remove` (const _Tp &__val)
- template<typename _Pred>
void `remove_if` (_Pred __pred)
- void `resize` (size_type __sz)
- void `resize` (size_type __sz, const value_type &__val)
- void `reverse` () noexcept
- void `sort` ()
- template<typename _Comp>
void `sort` (_Comp __comp)
- void `splice_after` (const_iterator __pos, forward_list &&__list) noexcept

- void **splice_after** (const_iterator __pos, forward_list &__list) noexcept
- void **splice_after** (const_iterator __pos, forward_list &&__list, const_iterator __i) noexcept
- void **splice_after** (const_iterator __pos, forward_list &__list, const_iterator __i) noexcept
- void **swap** (forward_list &__list) noexcept
- void **unique** ()
- template<typename _BinPred >
void **unique** (_BinPred __binary_pred)
- void **splice_after** (const_iterator __pos, forward_list &&, const_iterator __before, const_iterator __last) noexcept
- void **splice_after** (const_iterator __pos, forward_list &, const_iterator __before, const_iterator __last) noexcept

Private Member Functions

- template<typename... _Args>
_Node * **_M_create_node** (_Args &&...__args)
- _Fwd_list_node_base * **_M_erase_after** (_Fwd_list_node_base * __pos)
- _Fwd_list_node_base * **_M_erase_after** (_Fwd_list_node_base * __pos, _Fwd_list_node_base * __last)
- _Node * **_M_get_node** ()
- _Node_alloc_type & **_M_get_Node_allocator** () noexcept
- const _Node_alloc_type & **_M_get_Node_allocator** () const noexcept
- template<typename... _Args>
_Fwd_list_node_base * **_M_insert_after** (const_iterator __pos, _Args &&...__args)
- void **_M_put_node** (_Node * __p)

Private Attributes

- _Fwd_list_impl **_M_impl**

5.720.1 Detailed Description

```
template<typename _Tp, typename _Alloc = allocator<_Tp>>
class std::forward_list<_Tp, _Alloc>
```

A standard container with linear time access to elements, and fixed time insertion/deletion at any point in the sequence.

Template Parameters

<code>_Tp</code>	Type of element.
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_Tp></code> .

Meets the requirements of a [container](#), a [sequence](#), including the [optional sequence requirements](#) with the exception of `at` and `operator[]`.

This is a *singly linked* list. Traversal up the list requires linear time, but adding and removing elements (or *nodes*) is done in constant time, regardless of where the change takes place. Unlike `std::vector` and `std::deque`, random-access

iterators are not provided, so subscripting (`[]`) access is not allowed. For algorithms which only need sequential access, this lack makes no difference.

Also unlike the other standard containers, `std::forward_list` provides specialized algorithms unique to linked lists, such as splicing, sorting, and in-place reversal.

Definition at line 409 of file `forward_list.h`.

5.720.2 Constructor & Destructor Documentation

5.720.2.1 `template<typename _Tp, typename _Alloc = allocator<_Tp>> std::forward_list< _Tp, _Alloc >::forward_list ()`
`[inline], [noexcept]`

Creates a `forward_list` with no elements.

Definition at line 439 of file `forward_list.h`.

5.720.2.2 `template<typename _Tp, typename _Alloc = allocator<_Tp>> std::forward_list< _Tp, _Alloc >::forward_list (`
`const _Alloc & __al) [inline], [explicit], [noexcept]`

Creates a `forward_list` with no elements.

Parameters

<code>__al</code>	An allocator object.
-------------------	----------------------

Definition at line 449 of file `forward_list.h`.

5.720.2.3 `template<typename _Tp, typename _Alloc = allocator<_Tp>> std::forward_list< _Tp, _Alloc >::forward_list (`
`const forward_list< _Tp, _Alloc > & __list, const _Alloc & __al) [inline]`

Copy constructor with allocator argument.

Parameters

<code>__list</code>	Input list to copy.
<code>__al</code>	An allocator object.

Definition at line 459 of file `forward_list.h`.

References `std::forward_list< _Tp, _Alloc >::begin()`, and `std::forward_list< _Tp, _Alloc >::end()`.

5.720.2.4 `template<typename _Tp, typename _Alloc = allocator<_Tp>> std::forward_list< _Tp, _Alloc >::forward_list (`
`forward_list< _Tp, _Alloc > && __list, const _Alloc & __al) [inline], [noexcept]`

Move constructor with allocator argument.

Parameters

<code>__list</code>	Input list to move.
<code>__al</code>	An allocator object.

Definition at line 468 of file `forward_list.h`.

5.720.2.5 `template<typename _Tp, typename _Alloc = allocator<_Tp>> std::forward_list<_Tp, _Alloc>::forward_list (size_type __n, const _Alloc & __al = _Alloc()) [inline],[explicit]`

Creates a `forward_list` with default constructed elements.

Parameters

<code>__n</code>	The number of elements to initially create.
<code>__al</code>	An allocator object.

This constructor creates the `forward_list` with `__n` default constructed elements.

Definition at line 488 of file `forward_list.h`.

5.720.2.6 `template<typename _Tp, typename _Alloc = allocator<_Tp>> std::forward_list<_Tp, _Alloc>::forward_list (size_type __n, const _Tp & __value, const _Alloc & __al = _Alloc()) [inline]`

Creates a `forward_list` with copies of an exemplar element.

Parameters

<code>__n</code>	The number of elements to initially create.
<code>__value</code>	An element to copy.
<code>__al</code>	An allocator object.

This constructor fills the `forward_list` with `__n` copies of `__value`.

Definition at line 501 of file `forward_list.h`.

5.720.2.7 `template<typename _Tp, typename _Alloc = allocator<_Tp>> template<typename _InputIterator, typename = std::RequireInputIter<_InputIterator>> std::forward_list<_Tp, _Alloc>::forward_list (_InputIterator __first, _InputIterator __last, const _Alloc & __al = _Alloc()) [inline]`

Builds a `forward_list` from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__al</code>	An allocator object.

Create a forward_list consisting of copies of the elements from [*__first*,*__last*). This is linear in N (where N is distance(*__first*,*__last*)).

Definition at line 518 of file forward_list.h.

```
5.720.2.8  template<typename _Tp, typename _Alloc = allocator<_Tp>> std::forward_list< _Tp, _Alloc >::forward_list (
            const forward_list< _Tp, _Alloc > & __list )    [inline]
```

The forward_list copy constructor.

Parameters

<i>__list</i>	A forward_list of identical element and allocator types.
---------------	--

Definition at line 528 of file forward_list.h.

References std::forward_list< _Tp, _Alloc >::begin(), and std::forward_list< _Tp, _Alloc >::end().

```
5.720.2.9  template<typename _Tp, typename _Alloc = allocator<_Tp>> std::forward_list< _Tp, _Alloc >::forward_list (
            forward_list< _Tp, _Alloc > && __list )    [inline],[noexcept]
```

The forward_list move constructor.

Parameters

<i>__list</i>	A forward_list of identical element and allocator types.
---------------	--

The newly-created forward_list contains the exact contents of *__list*. The contents of *__list* are a valid, but unspecified forward_list.

Definition at line 542 of file forward_list.h.

```
5.720.2.10 template<typename _Tp, typename _Alloc = allocator<_Tp>> std::forward_list< _Tp, _Alloc >::forward_list (
            std::initializer_list<_Tp> __il, const _Alloc & __al = _Alloc() )    [inline]
```

Builds a forward_list from an initializer_list.

Parameters

<i>__il</i>	An initializer_list of value_type.
<i>__al</i>	An allocator object.

Create a forward_list consisting of copies of the elements in the initializer_list *__il*. This is linear in *__il*.size().

Definition at line 553 of file forward_list.h.

```
5.720.2.11 template<typename _Tp, typename _Alloc = allocator<_Tp>> std::forward_list<_Tp, _Alloc>::~~forward_list (
    ) [inline], [noexcept]
```

The forward_list dtor.

Definition at line 561 of file forward_list.h.

5.720.3 Member Function Documentation

```
5.720.3.1 template<typename _Tp, typename _Alloc = allocator<_Tp>> template<typename _InputIterator, typename =
    std::RequireInputIter<_InputIterator>>> void std::forward_list<_Tp, _Alloc>::assign ( _InputIterator __first,
    _InputIterator __last ) [inline]
```

Assigns a range to a forward_list.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

This function fills a forward_list with copies of the elements in the range [`__first`, `__last`).

Note that the assignment completely changes the forward_list and that the number of elements of the resulting forward_list is the same as the number of elements assigned. Old data is lost.

Definition at line 625 of file forward_list.h.

```
5.720.3.2 template<typename _Tp, typename _Alloc = allocator<_Tp>> void std::forward_list<_Tp, _Alloc>::assign (
    size_type __n, const _Tp & __val ) [inline]
```

Assigns a given value to a forward_list.

Parameters

<code>__n</code>	Number of elements to be assigned.
<code>__val</code>	Value to be assigned.

This function fills a forward_list with `__n` copies of the given value. Note that the assignment completely changes the forward_list, and that the resulting forward_list has `__n` elements. Old data is lost.

Definition at line 642 of file forward_list.h.

```
5.720.3.3 template<typename _Tp, typename _Alloc = allocator<_Tp>> void std::forward_list<_Tp, _Alloc>::assign (
    std::initializer_list<_Tp> __il ) [inline]
```

Assigns an initializer_list to a forward_list.

Parameters

<code>__il</code>	An initializer_list of value_type.
-------------------	------------------------------------

Replace the contents of the forward_list with copies of the elements in the initializer_list `__il`. This is linear in `il.size()`.

Definition at line 654 of file forward_list.h.

5.720.3.4 `template<typename _Tp, typename _Alloc = allocator<_Tp>> iterator std::forward_list< _Tp, _Alloc >::before_begin () [inline], [noexcept]`

Returns a read/write iterator that points before the first element in the forward_list. Iteration is done in ordinary element order.

Definition at line 669 of file forward_list.h.

Referenced by `std::forward_list< _Tp, _Alloc >::insert_after()`, and `std::forward_list< _Tp, _Alloc >::splice_after()`.

5.720.3.5 `template<typename _Tp, typename _Alloc = allocator<_Tp>> const_iterator std::forward_list< _Tp, _Alloc >::before_begin () const [inline], [noexcept]`

Returns a read-only (constant) iterator that points before the first element in the forward_list. Iteration is done in ordinary element order.

Definition at line 678 of file forward_list.h.

5.720.3.6 `template<typename _Tp, typename _Alloc = allocator<_Tp>> iterator std::forward_list< _Tp, _Alloc >::begin () [inline], [noexcept]`

Returns a read/write iterator that points to the first element in the forward_list. Iteration is done in ordinary element order.

Definition at line 686 of file forward_list.h.

Referenced by `std::forward_list< _Tp, _Alloc >::forward_list()`, and `std::forward_list< _Tp, _Alloc >::reverse()`.

5.720.3.7 `template<typename _Tp, typename _Alloc = allocator<_Tp>> const_iterator std::forward_list< _Tp, _Alloc >::begin () const [inline], [noexcept]`

Returns a read-only (constant) iterator that points to the first element in the forward_list. Iteration is done in ordinary element order.

Definition at line 695 of file forward_list.h.

5.720.3.8 `template<typename _Tp, typename _Alloc = allocator<_Tp>> const_iterator std::forward_list< _Tp, _Alloc >::cbefore_begin () const [inline], [noexcept]`

Returns a read-only (constant) iterator that points before the first element in the forward_list. Iteration is done in ordinary element order.

Definition at line 731 of file forward_list.h.

5.720.3.9 `template<typename _Tp, typename _Alloc = allocator<_Tp>> const_iterator std::forward_list<_Tp, _Alloc>::cbegin () const [inline], [noexcept]`

Returns a read-only (constant) iterator that points to the first element in the `forward_list`. Iteration is done in ordinary element order.

Definition at line 722 of file `forward_list.h`.

Referenced by `std::operator<()`, `std::forward_list<_Tp, _Alloc>::operator=()`, and `std::operator==(())`.

5.720.3.10 `template<typename _Tp, typename _Alloc = allocator<_Tp>> const_iterator std::forward_list<_Tp, _Alloc>::cend () const [inline], [noexcept]`

Returns a read-only (constant) iterator that points one past the last element in the `forward_list`. Iteration is done in ordinary element order.

Definition at line 740 of file `forward_list.h`.

Referenced by `std::operator<()`, `std::forward_list<_Tp, _Alloc>::operator=()`, and `std::operator==(())`.

5.720.3.11 `template<typename _Tp, typename _Alloc = allocator<_Tp>> void std::forward_list<_Tp, _Alloc>::clear () [inline], [noexcept]`

Erases all the elements.

Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1042 of file `forward_list.h`.

5.720.3.12 `template<typename _Tp, typename _Alloc = allocator<_Tp>> template<typename... _Args> iterator std::forward_list<_Tp, _Alloc>::emplace_after (const_iterator __pos, _Args &&... __args) [inline]`

Constructs object in `forward_list` after the specified iterator.

Parameters

<code>__pos</code>	A <code>const_iterator</code> into the <code>forward_list</code> .
<code>__args</code>	Arguments.

Returns

An iterator that points to the inserted data.

This function will insert an object of type `T` constructed with `T(std::forward<Args>(args)...) after the specified location. Due to the nature of a forward_list this operation can be done in constant time, and does not invalidate iterators and references.`

Definition at line 853 of file `forward_list.h`.

5.720.3.13 `template<typename _Tp, typename _Alloc = allocator<_Tp>> template<typename... _Args> void
std::forward_list< _Tp, _Alloc >::emplace_front (_Args &&... __args) [inline]`

Constructs object in forward_list at the front of the list.

Parameters

<code>__args</code>	Arguments.
---------------------	------------

This function will insert an object of type `Tp` constructed with `Tp(std::forward<Args>(args)...) at the front of the list`. Due to the nature of a `forward_list` this operation can be done in constant time, and does not invalidate iterators and references.

Definition at line 797 of file `forward_list.h`.

5.720.3.14 `template<typename _Tp, typename _Alloc = allocator<_Tp>> bool std::forward_list< _Tp, _Alloc >::empty ()
const [inline], [noexcept]`

Returns true if the `forward_list` is empty. (Thus `begin()` would equal `end()`.)

Definition at line 748 of file `forward_list.h`.

Referenced by `std::forward_list< _Tp, _Alloc >::insert_after()`, and `std::forward_list< _Tp, _Alloc >::splice_after()`.

5.720.3.15 `template<typename _Tp, typename _Alloc = allocator<_Tp>> iterator std::forward_list< _Tp, _Alloc >::end ()
[inline], [noexcept]`

Returns a read/write iterator that points one past the last element in the `forward_list`. Iteration is done in ordinary element order.

Definition at line 704 of file `forward_list.h`.

Referenced by `std::forward_list< _Tp, _Alloc >::forward_list()`, `std::forward_list< _Tp, _Alloc >::insert_after()`, `std::forward_list< _Tp, _Alloc >::reverse()`, and `std::forward_list< _Tp, _Alloc >::splice_after()`.

5.720.3.16 `template<typename _Tp, typename _Alloc = allocator<_Tp>> const_iterator std::forward_list< _Tp, _Alloc
>::end () const [inline], [noexcept]`

Returns a read-only iterator that points one past the last element in the `forward_list`. Iteration is done in ordinary element order.

Definition at line 713 of file `forward_list.h`.

5.720.3.17 `template<typename _Tp, typename _Alloc = allocator<_Tp>> iterator std::forward_list< _Tp, _Alloc
>::erase_after (const_iterator __pos) [inline]`

Removes the element pointed to by the iterator following `pos`.

Parameters

<code>__pos</code>	Iterator pointing before element to be erased.
--------------------	--

Returns

An iterator pointing to the element following the one that was erased, or `end()` if no such element exists.

This function will erase the element at the given position and thus shorten the `forward_list` by one.

Due to the nature of a `forward_list` this operation can be done in constant time, and only invalidates iterators/references to the element being removed. The user is also cautioned that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 956 of file `forward_list.h`.

```
5.720.3.18  template<typename _Tp, typename _Alloc = allocator<_Tp>> iterator std::forward_list<_Tp, _Alloc>
              >::erase_after( const_iterator __pos, const_iterator __last ) [inline]
```

Remove a range of elements.

Parameters

<code>__pos</code>	Iterator pointing before the first element to be erased.
<code>__last</code>	Iterator pointing to one past the last element to be erased.

Returns

@ `__last`.

This function will erase the elements in the range (`__pos`,`__last`) and shorten the `forward_list` accordingly.

This operation is linear time in the size of the range and only invalidates iterators/references to the element being removed. The user is also cautioned that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 979 of file `forward_list.h`.

```
5.720.3.19  template<typename _Tp, typename _Alloc = allocator<_Tp>> reference std::forward_list<_Tp, _Alloc>::front ( )
              [inline]
```

Returns a read/write reference to the data at the first element of the `forward_list`.

Definition at line 765 of file `forward_list.h`.

```
5.720.3.20  template<typename _Tp, typename _Alloc = allocator<_Tp>> const_reference std::forward_list<_Tp, _Alloc>
              >::front ( ) const [inline]
```

Returns a read-only (constant) reference to the data at the first element of the `forward_list`.

Definition at line 776 of file `forward_list.h`.

5.720.3.21 `template<typename _Tp, typename _Alloc = allocator<_Tp>> allocator_type std::forward_list< _Tp, _Alloc >::get_allocator () const [inline], [noexcept]`

Get a copy of the memory allocation object.

Definition at line 659 of file forward_list.h.

5.720.3.22 `template<typename _Tp, typename _Alloc = allocator<_Tp>> iterator std::forward_list< _Tp, _Alloc >::insert_after (const_iterator __pos, const _Tp & __val) [inline]`

Inserts given value into forward_list after specified iterator.

Parameters

<code>__pos</code>	An iterator into the forward_list.
<code>__val</code>	Data to be inserted.

Returns

An iterator that points to the inserted data.

This function will insert a copy of the given value after the specified location. Due to the nature of a forward_list this operation can be done in constant time, and does not invalidate iterators and references.

Definition at line 870 of file forward_list.h.

Referenced by std::forward_list< _Tp, _Alloc >::insert_after(), and std::forward_list< _Tp, _Alloc >::splice_after().

5.720.3.23 `template<typename _Tp, typename _Alloc > forward_list< _Tp, _Alloc >::iterator forward_list::insert_after (const_iterator __pos, size_type __n, const _Tp & __val)`

Inserts a number of copies of given data into the forward_list.

Parameters

<code>__pos</code>	An iterator into the forward_list.
<code>__n</code>	Number of elements to be inserted.
<code>__val</code>	Data to be inserted.

Returns

An iterator pointing to the last inserted copy of *val* or *pos* if *n* == 0.

This function will insert a specified number of copies of the given data after the location specified by *pos*.

This operation is linear in the number of elements inserted and does not invalidate iterators and references.

Definition at line 260 of file forward_list.tcc.

References std::forward_list< _Tp, _Alloc >::before_begin(), std::forward_list< _Tp, _Alloc >::end(), and std::forward_list< _Tp, _Alloc >::insert_after().

5.720.3.24 `template<typename _Tp, typename _Alloc > template<typename _InputIterator, typename > forward_list< _Tp, _Alloc >::iterator forward_list::insert_after(const_iterator __pos, _InputIterator __first, _InputIterator __last)`

Inserts a range into the `forward_list`.

Parameters

<code>__pos</code>	An iterator into the <code>forward_list</code> .
<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Returns

An iterator pointing to the last inserted element or `__pos` if `__first == __last`.

This function will insert copies of the data in the range `[__first, __last)` into the `forward_list` after the location specified by `__pos`.

This operation is linear in the number of elements inserted and does not invalidate iterators and references.

Definition at line 275 of file `forward_list.tcc`.

References `std::forward_list< _Tp, _Alloc >::before_begin()`, `std::forward_list< _Tp, _Alloc >::empty()`, `std::forward_list< _Tp, _Alloc >::end()`, and `std::forward_list< _Tp, _Alloc >::remove()`.

5.720.3.25 `template<typename _Tp, typename _Alloc = allocator<_Tp>> iterator std::forward_list< _Tp, _Alloc >::insert_after(const_iterator __pos, std::initializer_list<_Tp> __il) [inline]`

Inserts the contents of an `initializer_list` into `forward_list` after the specified iterator.

Parameters

<code>__pos</code>	An iterator into the <code>forward_list</code> .
<code>__il</code>	An <code>initializer_list</code> of value_type.

Returns

An iterator pointing to the last inserted element or `__pos` if `__il` is empty.

This function will insert copies of the data in the `initializer_list` `__il` into the `forward_list` before the location specified by `__pos`.

This operation is linear in the number of elements inserted and does not invalidate iterators and references.

Definition at line 935 of file `forward_list.h`.

5.720.3.26 `template<typename _Tp, typename _Alloc = allocator<_Tp>> size_type std::forward_list< _Tp, _Alloc >::max_size () const [inline], [noexcept]`

Returns the largest possible number of elements of forward_list.

Definition at line 755 of file forward_list.h.

5.720.3.27 `template<typename _Tp, typename _Alloc = allocator<_Tp>> void std::forward_list< _Tp, _Alloc >::merge (forward_list< _Tp, _Alloc > && __list) [inline]`

Merge sorted lists.

Parameters

<code>__list</code>	Sorted list to merge.
---------------------	-----------------------

Assumes that both `list` and this list are sorted according to operator<(). Merges elements of `__list` into this list in sorted order, leaving `__list` empty when complete. Elements in this list precede elements in `__list` that are equal.

Definition at line 1182 of file forward_list.h.

Referenced by std::forward_list< _Tp, _Alloc >::unique().

5.720.3.28 `template<typename _Tp, typename _Alloc > template<typename _Comp > void forward_list::merge (forward_list< _Tp, _Alloc > && __list, _Comp __comp)`

Merge sorted lists according to comparison function.

Parameters

<code>__list</code>	Sorted list to merge.
<code>__comp</code>	Comparison function defining sort order.

Assumes that both `__list` and this list are sorted according to `comp`. Merges elements of `__list` into this list in sorted order, leaving `__list` empty when complete. Elements in this list precede elements in `__list` that are equivalent according to `comp()`.

Definition at line 353 of file forward_list.tcc.

5.720.3.29 `template<typename _Tp, typename _Alloc > forward_list< _Tp, _Alloc > & forward_list::operator= (const forward_list< _Tp, _Alloc > & __list)`

The forward_list assignment operator.

Parameters

<code>__list</code>	A forward_list of identical element and allocator types.
---------------------	--

All the elements of `__list` are copied, but unlike the copy constructor, the allocator object is not copied.

Definition at line 144 of file `forward_list.tcc`.

References `std::__addressof()`, `std::forward_list<_Tp, _Alloc>::cbegin()`, `std::forward_list<_Tp, _Alloc>::cend()`, and `std::forward_list<_Tp, _Alloc>::resize()`.

5.720.3.30 `template<typename _Tp, typename _Alloc = allocator<_Tp>> forward_list& std::forward_list<_Tp, _Alloc>::operator= (forward_list<_Tp, _Alloc> && __list) [inline], [noexcept]`

The `forward_list` move assignment operator.

Parameters

<code>__list</code>	A <code>forward_list</code> of identical element and allocator types.
---------------------	---

The contents of `__list` are moved into this `forward_list` (without copying, if the allocators permit it). `__list` is a valid, but unspecified `forward_list`

Definition at line 585 of file `forward_list.h`.

5.720.3.31 `template<typename _Tp, typename _Alloc = allocator<_Tp>> forward_list& std::forward_list<_Tp, _Alloc>::operator= (std::initializer_list<_Tp> __il) [inline]`

The `forward_list` initializer list assignment operator.

Parameters

<code>__il</code>	An <code>initializer_list</code> of <code>value_type</code> .
-------------------	---

Replace the contents of the `forward_list` with copies of the elements in the `initializer_list __il`. This is linear in `__il.size()`.

Definition at line 604 of file `forward_list.h`.

5.720.3.32 `template<typename _Tp, typename _Alloc = allocator<_Tp>> void std::forward_list<_Tp, _Alloc>::pop_front () [inline]`

Removes first element.

This is a typical stack operation. It shrinks the `forward_list` by one. Due to the nature of a `forward_list` this operation can be done in constant time, and only invalidates iterators/references to the element being removed.

Note that no data is returned, and if the first element's data is needed, it should be retrieved before `pop_front()` is called.

Definition at line 835 of file `forward_list.h`.

5.720.3.33 `template<typename _Tp, typename _Alloc = allocator<_Tp>> void std::forward_list<_Tp, _Alloc>::push_front (const _Tp & __val) [inline]`

Add data to the front of the `forward_list`.

Parameters

<code>__val</code>	Data to be added.
--------------------	-------------------

This is a typical stack operation. The function creates an element at the front of the forward_list and assigns the given data to it. Due to the nature of a forward_list this operation can be done in constant time, and does not invalidate iterators and references.

Definition at line 812 of file forward_list.h.

5.720.334 `template<typename _Tp, typename _Alloc > void forward_list::remove (const _Tp & __val)`

Remove all elements equal to value.

Parameters

<code>__val</code>	The value to remove.
--------------------	----------------------

Removes every element in the list equal to `__val`. Remaining elements stay in list order. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 288 of file forward_list.tcc.

References `std::__addressof()`, and `std::forward_list< _Tp, _Alloc >::remove_if()`.

Referenced by `std::forward_list< _Tp, _Alloc >::insert_after()`.

5.720.335 `template<typename _Tp, typename _Alloc > template<typename _Pred > void forward_list::remove_if (_Pred __pred)`

Remove all elements satisfying a predicate.

Parameters

<code>__pred</code>	Unary predicate function or object.
---------------------	-------------------------------------

Removes every element in the list for which the predicate returns true. Remaining elements stay in list order. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 316 of file forward_list.tcc.

References `std::forward_list< _Tp, _Alloc >::unique()`.

Referenced by `std::forward_list< _Tp, _Alloc >::remove()`.

5.720.336 `template<typename _Tp, typename _Alloc > void forward_list::resize (size_type __sz)`

Resizes the forward_list to the specified number of elements.

Parameters

<code>__sz</code>	Number of elements the <code>forward_list</code> should contain.
-------------------	--

This function will resize the `forward_list` to the specified number of elements. If the number is smaller than the `forward_list`'s current number of elements the `forward_list` is truncated, otherwise the `forward_list` is extended and the new elements are default constructed.

Definition at line 186 of file `forward_list.tcc`.

References `std::end()`.

Referenced by `std::forward_list<_Tp, _Alloc>::operator=()`.

5.720.3.37 `template<typename _Tp, typename _Alloc> void forward_list::resize (size_type __sz, const value_type & __val)`

Resizes the `forward_list` to the specified number of elements.

Parameters

<code>__sz</code>	Number of elements the <code>forward_list</code> should contain.
<code>__val</code>	Data with which new elements should be populated.

This function will resize the `forward_list` to the specified number of elements. If the number is smaller than the `forward_list`'s current number of elements the `forward_list` is truncated, otherwise the `forward_list` is extended and new elements are populated with given data.

Definition at line 205 of file `forward_list.tcc`.

References `std::end()`, and `std::forward_list<_Tp, _Alloc>::splice_after()`.

5.720.3.38 `template<typename _Tp, typename _Alloc = allocator<_Tp>> void std::forward_list<_Tp, _Alloc>::reverse ()`
`[inline], [noexcept]`

Reverse the elements in list.

Reverse the order of elements in the list in linear time.

Definition at line 1235 of file `forward_list.h`.

References `std::begin()`, `std::forward_list<_Tp, _Alloc>::begin()`, `std::end()`, and `std::forward_list<_Tp, _Alloc>::end()`.

5.720.3.39 `template<typename _Tp, typename _Alloc = allocator<_Tp>> void std::forward_list<_Tp, _Alloc>::sort ()`
`[inline]`

Sort the elements of the list.

Sorts the elements of this list in NlogN time. Equivalent elements remain in list order.

Definition at line 1216 of file `forward_list.h`.

Referenced by `std::operator==()`.

5.720.3.40 `template<typename _Tp, class _Alloc> template<typename _Comp> void forward_list::sort (_Comp __comp)`

Sort the forward_list using a comparison function.

Sorts the elements of this list in NlogN time. Equivalent elements remain in list order.

Definition at line 399 of file forward_list.tcc.

5.720.3.41 `template<typename _Tp, typename _Alloc = allocator<_Tp>> void std::forward_list<_Tp, _Alloc>::splice_after (const_iterator __pos, forward_list<_Tp, _Alloc> && __list) [inline], [noexcept]`

Insert contents of another forward_list.

Parameters

<code>__pos</code>	Iterator referencing the element to insert after.
<code>__list</code>	Source list.

The elements of *list* are inserted in constant time after the element referenced by *pos*. *list* becomes an empty list.

Requires this != x.

Definition at line 1059 of file forward_list.h.

References std::forward_list< _Tp, _Alloc >::before_begin(), std::forward_list< _Tp, _Alloc >::empty(), and std::forward_list< _Tp, _Alloc >::end().

Referenced by std::forward_list< _Tp, _Alloc >::resize().

5.720.3.42 `template<typename _Tp, typename _Alloc> void forward_list::splice_after (const_iterator __pos, forward_list<_Tp, _Alloc> && __list, const_iterator __i) [noexcept]`

Insert element from another forward_list.

Parameters

<code>__pos</code>	Iterator referencing the element to insert after.
<code>__list</code>	Source list.
<code>__i</code>	Iterator referencing the element before the element to move.

Removes the element in list *list* referenced by *i* and inserts it into the current list after *pos*.

Definition at line 243 of file forward_list.tcc.

References std::forward_list< _Tp, _Alloc >::insert_after().

5.720.3.43 `template<typename _Tp, typename _Alloc = allocator<_Tp>> void std::forward_list<_Tp, _Alloc>::splice_after (`
`const_iterator __pos, forward_list<_Tp, _Alloc> &&, const_iterator __before, const_iterator __last)`
`[inline], [noexcept]`

Insert range from another forward_list.

Parameters

<code>__pos</code>	Iterator referencing the element to insert after.
<code>__list</code>	Source list.
<code>__before</code>	Iterator referencing before the start of range in list.
<code>__last</code>	Iterator referencing the end of range in list.

Removes elements in the range (`__before`,`__last`) and inserts them after `__pos` in constant time.

Undefined if `__pos` is in (`__before`,`__last`).

Definition at line 1103 of file `forward_list.h`.

```
5.720.3.44 template<typename _Tp, typename _Alloc = allocator<_Tp>> void std::forward_list<_Tp, _Alloc>::splice_after (
    const_iterator __pos, forward_list<_Tp, _Alloc> &, const_iterator __before, const_iterator __last )
    [inline], [noexcept]
```

Insert range from another `forward_list`.

Parameters

<code>__pos</code>	Iterator referencing the element to insert after.
<code>__list</code>	Source list.
<code>__before</code>	Iterator referencing before the start of range in list.
<code>__last</code>	Iterator referencing the end of range in list.

Removes elements in the range (`__before`,`__last`) and inserts them after `__pos` in constant time.

Undefined if `__pos` is in (`__before`,`__last`).

Definition at line 1108 of file `forward_list.h`.

```
5.720.3.45 template<typename _Tp, typename _Alloc = allocator<_Tp>> void std::forward_list<_Tp, _Alloc>::swap (
    forward_list<_Tp, _Alloc> & __list ) [inline], [noexcept]
```

Swaps data with another `forward_list`.

Parameters

<code>__list</code>	A <code>forward_list</code> of the same element and allocator types.
---------------------	--

This exchanges the elements between two lists in constant time. Note that the global `std::swap()` function is specialized such that `std::swap(l1,l2)` will feed to this function.

Definition at line 996 of file `forward_list.h`.

5.720.3.46 `template<typename _Tp, typename _Alloc = allocator<_Tp>> void std::forward_list<_Tp, _Alloc>::unique ()`
`[inline]`

Remove consecutive duplicate elements.

For each consecutive set of elements with the same value, remove all but the first one. Remaining elements stay in list order. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1153 of file `forward_list.h`.

Referenced by `std::forward_list<_Tp, _Alloc>::remove_if()`.

5.720.3.47 `template<typename _Tp, typename _Alloc> template<typename _BinPred> void forward_list::unique (_BinPred`
`__binary_pred)`

Remove consecutive elements satisfying a predicate.

Parameters

<code>__binary_pred</code>	Binary predicate function or object.
----------------------------	--------------------------------------

For each consecutive set of elements `[first,last)` that satisfy `predicate(first,i)` where `i` is an iterator in `[first,last)`, remove all but the first one. Remaining elements stay in list order. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 332 of file `forward_list.tcc`.

References `std::begin()`, `std::end()`, and `std::forward_list<_Tp, _Alloc>::merge()`.

The documentation for this class was generated from the following files:

- [forward_list.h](#)
- [forward_list.tcc](#)

5.721 `std::fpos<_StateT>` Class Template Reference

Public Member Functions

- [fpos](#) ([streamoff](#) __off)
- [operator streamoff](#) () const
- [fpos operator+](#) ([streamoff](#) __off) const
- [fpos & operator+=](#) ([streamoff](#) __off)
- [fpos operator-](#) ([streamoff](#) __off) const
- [streamoff operator-](#) (const [fpos](#) &__other) const
- [fpos & operator-=](#) ([streamoff](#) __off)
- void [state](#) (_StateT __st)
- _StateT [state](#) () const

5.721.1 Detailed Description

```
template<typename _StateT>
class std::fpos<_StateT>
```

Class representing stream positions.

The standard places no requirements upon the template parameter `StateT`. In this implementation `StateT` must be `DefaultConstructible`, `CopyConstructible` and `Assignable`. The standard only requires that `fpos` should contain a member of type `StateT`. In this implementation it also contains an offset stored as a signed integer.

Parameters

<i>StateT</i>	Type passed to and returned from <code>state()</code> .
---------------	---

Definition at line 112 of file `postypes.h`.

5.721.2 Constructor & Destructor Documentation

5.721.2.1 `template<typename _StateT> std::fpos<_StateT>::fpos (streamoff __off) [inline]`

Construct position from offset.

Definition at line 133 of file `postypes.h`.

5.721.3 Member Function Documentation

5.721.3.1 `template<typename _StateT> std::fpos<_StateT>::operator streamoff () const [inline]`

Convert to `streamoff`.

Definition at line 137 of file `postypes.h`.

5.721.3.2 `template<typename _StateT> fpos std::fpos<_StateT>::operator+ (streamoff __off) const [inline]`

Add position and offset.

Definition at line 178 of file `postypes.h`.

5.721.3.3 `template<typename _StateT> fpos& std::fpos<_StateT>::operator+= (streamoff __off) [inline]`

Add offset to this position.

Definition at line 154 of file `postypes.h`.

5.721.3.4 `template<typename _StateT> fpos std::fpos<_StateT>::operator-(streamoff __off) const [inline]`

Subtract offset from position.

Definition at line 192 of file postypes.h.

5.721.3.5 `template<typename _StateT> streamoff std::fpos<_StateT>::operator-(const fpos<_StateT> &__other) const [inline]`

Subtract position to return offset.

Definition at line 205 of file postypes.h.

5.721.3.6 `template<typename _StateT> fpos& std::fpos<_StateT>::operator-=(streamoff __off) [inline]`

Subtract offset from this position.

Definition at line 165 of file postypes.h.

5.721.3.7 `template<typename _StateT> void std::fpos<_StateT>::state(_StateT __st) [inline]`

Remember the value of *st*.

Definition at line 141 of file postypes.h.

5.721.3.8 `template<typename _StateT> _StateT std::fpos<_StateT>::state() const [inline]`

Return the last set value of *st*.

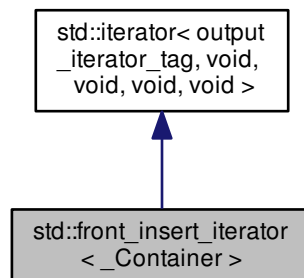
Definition at line 146 of file postypes.h.

The documentation for this class was generated from the following file:

- [postypes.h](#)

5.722 `std::front_insert_iterator<_Container>` Class Template Reference

Inheritance diagram for `std::front_insert_iterator<_Container>`:



Public Types

- typedef `_Container` `container_type`
- typedef void `difference_type`
- typedef `output_iterator_tag` `iterator_category`
- typedef void `pointer`
- typedef void `reference`
- typedef void `value_type`

Public Member Functions

- `front_insert_iterator` (`_Container` &__x)
- `front_insert_iterator` & `operator*` ()
- `front_insert_iterator` & `operator++` ()
- `front_insert_iterator` `operator++` (int)
- `front_insert_iterator` & `operator=` (const typename `_Container::value_type` &__value)
- `front_insert_iterator` & `operator=` (typename `_Container::value_type` &&__value)

Protected Attributes

- `_Container` * **`container`**

5.722.1 Detailed Description

```
template<typename _Container>
class std::front_insert_iterator<_Container>
```

Turns assignment into insertion.

These are output iterators, constructed from a container-of-T. Assigning a T to the iterator prepends it to the container using `push_front`.

Tip: Using the `front_inserter` function to create these iterators can save typing.

Definition at line 541 of file `bits/stl_iterator.h`.

5.722.2 Member Typedef Documentation

5.722.2.1 `template<typename _Container> typedef _Container std::front_insert_iterator<_Container>::container_type`

A nested typedef for the type of whatever container you used.

Definition at line 549 of file `bits/stl_iterator.h`.

5.722.2.2 `typedef void std::iterator< output_iterator_tag , void , void , void , void >::difference_type` [inherited]

Distance between iterators is represented as this type.

Definition at line 125 of file `stl_iterator_base_types.h`.

5.722.2.3 `typedef output_iterator_tag std::iterator< output_iterator_tag , void , void , void , void >::iterator_category` [inherited]

One of the [tag types](#).

Definition at line 121 of file `stl_iterator_base_types.h`.

5.722.2.4 `typedef void std::iterator< output_iterator_tag , void , void , void , void >::pointer` [inherited]

This type represents a pointer-to-value_type.

Definition at line 127 of file `stl_iterator_base_types.h`.

5.722.2.5 `typedef void std::iterator< output_iterator_tag , void , void , void , void >::reference` [inherited]

This type represents a reference-to-value_type.

Definition at line 129 of file `stl_iterator_base_types.h`.

5.722.2.6 `typedef void std::iterator< output_iterator_tag , void , void , void , void >::value_type` [inherited]

The type "pointed to" by the iterator.

Definition at line 123 of file `stl_iterator_base_types.h`.

5.722.3 Constructor & Destructor Documentation

5.722.3.1 `template<typename _Container > std::front_insert_iterator< _Container >::front_insert_iterator (_Container &_x)` [inline],[explicit]

The only way to create this iterator is with a container.

Definition at line 552 of file `bits/stl_iterator.h`.

5.722.4 Member Function Documentation

5.722.4.1 `template<typename _Container > front_insert_iterator& std::front_insert_iterator< _Container >::operator* ()` [inline]

Simply returns `*this`.

Definition at line 591 of file `bits/stl_iterator.h`.

5.722.4.2 `template<typename _Container> front_insert_iterator& std::front_insert_iterator<_Container>::operator++ () [inline]`

Simply returns `*this`. (This iterator does not *move*.)

Definition at line 596 of file `bits/stl_iterator.h`.

5.722.4.3 `template<typename _Container> front_insert_iterator std::front_insert_iterator<_Container>::operator++ (int) [inline]`

Simply returns `*this`. (This iterator does not *move*.)

Definition at line 601 of file `bits/stl_iterator.h`.

5.722.4.4 `template<typename _Container> front_insert_iterator& std::front_insert_iterator<_Container>::operator= (const typename _Container::value_type & __value) [inline]`

Parameters

<code>__value</code>	An instance of whatever type <code>container_type::const_reference</code> is; presumably a reference-to-const T for <code>container<T></code> .
----------------------	---

Returns

This iterator, for chained operations.

This kind of iterator doesn't really have a *position* in the container (you can think of the position as being permanently at the front, if you like). Assigning a value to the iterator will always prepend the value to the front of the container.

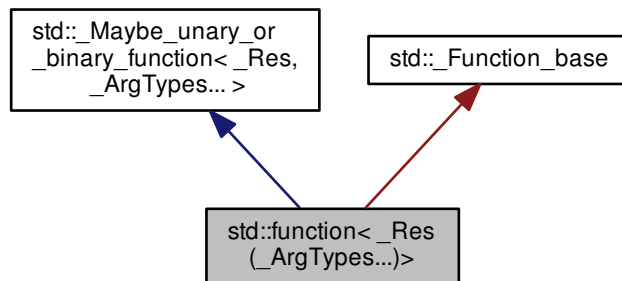
Definition at line 575 of file `bits/stl_iterator.h`.

The documentation for this class was generated from the following file:

- [bits/stl_iterator.h](#)

5.723 `std::function<_Res(_ArgTypes...)>` Class Template Reference

Inheritance diagram for `std::function<_Res(_ArgTypes...)>`:



Public Types

- typedef `_Res` **result_type**

Public Member Functions

- `function` () noexcept
- `function` (nullptr_t) noexcept
- `function` (const function &__x)
- `function` (function &&__x)
- template<typename _Functor, typename = _Requires<__not_<is_same<_Functor, function>>, void>, typename = _Requires<_↔Callable<_Functor>, void>>>
`function` (_Functor)
- `operator bool` () const noexcept
- `_Res operator()` (_ArgTypes...__args) const
- `function & operator=` (const function &__x)
- `function & operator=` (function &&__x)
- `function & operator=` (nullptr_t) noexcept
- template<typename _Functor >
`_Requires<_Callable<typename decay<_Functor>::type>, function &> operator=` (_Functor &&__f)
- template<typename _Functor >
`function & operator=` (reference_wrapper<_Functor> __f) noexcept
- void `swap` (function &__x) noexcept
- template<typename _Functor >
`_Functor * target` () noexcept
- template<typename _Functor >
`const _Functor * target` () const noexcept
- const `type_info & target_type` () const noexcept

Private Types

- typedef bool(* **_Manager_type**) (_Any_data &, const _Any_data &, _Manager_operation)

Private Member Functions

- bool **_M_empty** () const

Private Attributes

- _Any_data **_M_functor**
- _Manager_type **_M_manager**

Static Private Attributes

- static const std::size_t **_M_max_align**
- static const std::size_t **_M_max_size**

5.723.1 Detailed Description

```
template<typename _Res, typename... _ArgTypes>
class std::function< _Res(_ArgTypes...)>
```

Primary class template for std::function.

Polymorphic function wrapper.

Definition at line 1834 of file functional.

5.723.2 Constructor & Destructor Documentation

```
5.723.2.1 template<typename _Res, typename... _ArgTypes> std::function< _Res(_ArgTypes...)>::function ( ) [inline],
[noexcept]
```

Default construct creates an empty function call wrapper.

Postcondition

```
!(bool)*this
```

Definition at line 1861 of file functional.

```
5.723.2.2 template<typename _Res, typename... _ArgTypes> std::function< _Res(_ArgTypes...)>::function ( nullptr_t )
[inline], [noexcept]
```

Creates an empty function call wrapper.

Postcondition

```
!(bool)*this
```

Definition at line 1868 of file functional.

```
5.723.2.3 template<typename _Res, typename... _ArgTypes> std::function< _Res(_ArgTypes...)>::function ( const function<
_Res(_ArgTypes...)> & _x )
```

Function copy constructor.

Parameters

<code>__x</code>	A function object with identical call signature.
------------------	--

Postcondition

```
bool(*this) == bool(__x)
```

The newly-created function contains a copy of the target of `__x` (if it has one).

Definition at line 2093 of file functional.

```
5.723.2.4 template<typename _Res , typename... _ArgTypes> std::function< _Res(_ArgTypes...)>::function ( function<
    _Res(_ArgTypes...)> && __x ) [inline]
```

Function move constructor.

Parameters

<code>__x</code>	A function object rvalue with identical call signature.
------------------	---

The newly-created function contains the target of `__x` (if it has one).

Definition at line 1888 of file functional.

```
5.723.2.5 template<typename _Res , typename... _ArgTypes> template<typename _Functor , typename , typename >
    std::function< _Res(_ArgTypes...)>::function ( _Functor __f )
```

Builds a function that targets a copy of the incoming function object.

Parameters

<code>__f</code>	A function object that is callable with parameters of type T1, T2, ..., TN and returns a value convertible to Res.
------------------	--

The newly-created function object will target a copy of `__f`. If `__f` is `reference_wrapper<F>`, then this function object will contain a reference to the function object `__f.get()`. If `__f` is a NULL function pointer or NULL pointer-to-member, the newly-created object will be empty.

If `__f` is a non-NULL function pointer or an object of type `reference_wrapper<F>`, this function will not throw.

Definition at line 2107 of file functional.

5.723.3 Member Function Documentation

5.723.3.1 `template<typename _Res, typename... _ArgTypes> std::function< _Res(_ArgTypes...)>::operator bool () const`
`[inline], [explicit], [noexcept]`

Determine if the function wrapper has a target.

Returns

`true` when this function object contains a target, or `false` when it is empty.

This function will not throw an exception.

Definition at line 2042 of file functional.

5.723.3.2 `template<typename _Res, typename... _ArgTypes> _Res std::function< _Res(_ArgTypes...)>::operator() (_ArgTypes...
 __args) const`

Invokes the function targeted by `*this`.

Returns

the result of the target.

Exceptions

<code>bad_function_call</code>	when <code>!(bool)*this</code>
--------------------------------	--------------------------------

The function call operator invokes the target function object stored by `this`.

Definition at line 2123 of file functional.

5.723.3.3 `template<typename _Res, typename... _ArgTypes> function& std::function< _Res(_ArgTypes...)>::operator= (const
 function< _Res(_ArgTypes...)> &__x) [inline]`

Function assignment operator.

Parameters

<code>__x</code>	A function with identical call signature.
------------------	---

Postcondition

`(bool)*this == (bool)x`

Returns

`*this`

The target of `__x` is copied to `*this`. If `__x` has no target, then `*this` will be empty.

If `__x` targets a function pointer or a reference to a function object, then this operation will not throw an exception.

Definition at line 1929 of file functional.

5.723.3.4 `template<typename _Res, typename... _ArgTypes> function& std::function< _Res(_ArgTypes...)>::operator= (function< _Res(_ArgTypes...)> && __x) [inline]`

Function move-assignment operator.

Parameters

<code>__x</code>	A function rvalue with identical call signature.
------------------	--

Returns

`*this`

The target of `__x` is moved to `*this`. If `__x` has no target, then `*this` will be empty.

If `__x` targets a function pointer or a reference to a function object, then this operation will not throw an exception.

Definition at line 1947 of file functional.

5.723.3.5 `template<typename _Res, typename... _ArgTypes> function& std::function< _Res(_ArgTypes...)>::operator= (nullptr_t) [inline], [noexcept]`

Function assignment to zero.

Postcondition

`!(bool)*this`

Returns

`*this`





















































The target of `*this` is deallocated, leaving it empty.

Definition at line 1961 of file functional.

5.723.3.6 `template<typename _Res, typename... _ArgTypes> template<typename _Functor > _Requires< _Callable<typename decay<_Functor>::type>, function&> std::function< _Res(_ArgTypes...)>::operator= (_Functor && __f) [inline]`

Function assignment to a new target.

Parameters

	A function object that is callable with parameters of type T1, T2, ..., TN and returns a value convertible to Res.
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	

Returns

*this

This function object wrapper will target a copy of `__f`. If `__f` is `reference_wrapper<F>`, then this function object will contain a reference to the function object `__f.get()`. If `__f` is a NULL function pointer or NULL pointer-to-member, `this` object will be empty.

If `__f` is a non-NULL function pointer or an object of type `reference_wrapper<F>`, this function will not throw.

Definition at line 1990 of file functional.

5.723.3.7 `template<typename _Res, typename... _ArgTypes> template<typename _Functor> function& std::function<_Res(_ArgTypes...)>::operator=(reference_wrapper<_Functor> __f) [inline], [noexcept]`




















This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.

Definition at line 1999 of file functional.

5.723.3.8 `template<typename _Res, typename... _ArgTypes> void std::function<_Res(_ArgTypes...)>::swap (function<_Res(_ArgTypes...)> &__x) [inline], [noexcept]`

Swap the targets of two function objects.

Parameters

	A function with identical call signature.
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	
	

Swap the targets of `this` function object and `__f`. This function will not throw an exception.

Definition at line 2014 of file functional.

5.723.3.9 `template<typename _Res, typename... _ArgTypes> template<typename _Functor> _Functor * std::function<_Res(_ArgTypes...)>::target () [noexcept]`

Access the stored target function object.

Returns

Returns a pointer to the stored target function object, if `typeid(Functor).equals(target_type())`; otherwise, a NULL pointer.

This function will not throw an exception.

Definition at line 2150 of file functional.

```
5.723.3.10 template<typename _Res , typename... _ArgTypes> template<typename _Functor > const _Functor * std::function<
    _Res(_ArgTypes...)>::target ( ) const [noexcept]
```

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.

Definition at line 2169 of file functional.

```
5.723.3.11 template<typename _Res , typename... _ArgTypes> const type_info & std::function<
    _Res(_ArgTypes...)>::target_type ( ) const [noexcept]
```

Determine the type of the target of this function object wrapper.

Returns

the type identifier of the target function object, or `typeid(void)` if `!(bool)*this`.

This function will not throw an exception.

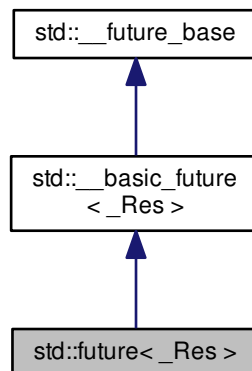
Definition at line 2134 of file functional.

The documentation for this class was generated from the following file:

- [functional](#)

5.724 std::future< _Res > Class Template Reference

Inheritance diagram for `std::future< _Res >`:



Public Types

- template<typename _Res>
using **_Ptr** = **unique_ptr**<_Res, _Result_base::_Deleter>
- using **_State_base** = _State_baseV2

Public Member Functions

- **future** (**future** &&__uf) noexcept
- **future** (const **future** &)=delete
- **_Res** **get** ()
- **future** & **operator=** (const **future** &)=delete
- **future** & **operator=** (**future** &&__fut) noexcept
- **shared_future**<_Res> **share** ()
- bool **valid** () const noexcept
- void **wait** () const
- template<typename _Rep, typename _Period>
future_status **wait_for** (const **chrono::duration**<_Rep, _Period> &__rel) const
- template<typename _Clock, typename _Duration>
future_status **wait_until** (const **chrono::time_point**<_Clock, _Duration> &__abs) const

Static Public Member Functions

- template<typename _Res, typename _Allocator>
static **_Ptr**<**_Result_alloc**<_Res, _Allocator>> **_S_allocate_result** (const _Allocator &__a)
- template<typename _Res, typename _Tp>
static **_Ptr**<**_Result**<_Res>> **_S_allocate_result** (const **std::allocator**<_Tp> &__a)
- template<typename _BoundFn>
static **std::shared_ptr**<_State_base> **_S_make_async_state** (_BoundFn &&__fn)
- template<typename _BoundFn>
static **std::shared_ptr**<_State_base> **_S_make_deferred_state** (_BoundFn &&__fn)
- template<typename _Res_ptr, typename _BoundFn>
static **_Task_setter**<_Res_ptr, _BoundFn> **_S_task_setter** (_Res_ptr &__ptr, _BoundFn &__call)

Protected Types

- typedef **__future_base::_Result**<_Res> & **__result_type**

Protected Member Functions

- **__result_type** **_M_get_result** () const
- void **_M_swap** (**__basic_future** &__that) noexcept

Friends

- template<typename _Fn, typename... _Args>
future<__async_result_of<_Fn, _Args...>> **async** (**launch**, _Fn &&, _Args &&...)
- template<typename >
class **packaged_task**
- class **promise**<_Res>

5.724.1 Detailed Description

```
template<typename _Res>
class std::future<_Res>
```

Primary template for future.

Definition at line 115 of file future.

5.724.2 Member Typedef Documentation

5.724.2.1 `template<typename _Res> using std::__future_base::Ptr = unique_ptr<_Res, _Result_base::Deleter>`
[inherited]

A unique_ptr for result objects.

Definition at line 214 of file future.

5.724.3 Constructor & Destructor Documentation

5.724.3.1 `template<typename _Res> std::future<_Res>::future (future<_Res> && __uf)` [inline],
[noexcept]

Move constructor.

Definition at line 749 of file future.

5.724.4 Member Function Documentation

5.724.4.1 `template<typename _Res> __result_type std::__basic_future<_Res>::M_get_result () const` [inline],
[protected], [inherited]

Wait for the state to be ready and rethrow any stored exception.

Definition at line 684 of file future.

5.724.4.2 `template<typename _Res> _Res std::future<_Res>::get ()` [inline]

Retrieving the value.

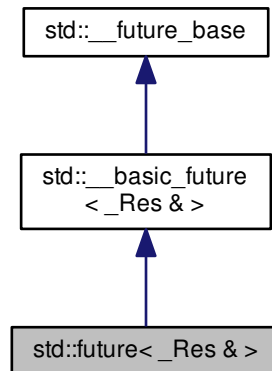
Definition at line 763 of file future.

The documentation for this class was generated from the following file:

- [future](#)

5.725 `std::future<_Res &>` Class Template Reference

Inheritance diagram for `std::future<_Res &>`:



Public Types

- `template<typename _Res >`
`using _Ptr = unique_ptr<_Res, _Result_base::_Deleter >`
- `using _State_base = _State_baseV2`

Public Member Functions

- `future (future &&__uf) noexcept`
- `future (const future &)=delete`
- `_Res & get ()`
- `future & operator= (const future &)=delete`
- `future & operator= (future &&__fut) noexcept`
- `shared_future<_Res &> share ()`
- `bool valid () const noexcept`
- `void wait () const`
- `future_status wait_for (const chrono::duration<_Rep, _Period > &__rel) const`
- `future_status wait_until (const chrono::time_point<_Clock, _Duration > &__abs) const`

Static Public Member Functions

- `template<typename _Res , typename _Allocator >`
`static _Ptr< _Result_alloc< _Res, _Allocator > > _S_allocate_result (const _Allocator &__a)`
- `template<typename _Res , typename _Tp >`
`static _Ptr< _Result< _Res > > _S_allocate_result (const std::allocator< _Tp > &__a)`
- `template<typename _BoundFn >`
`static std::shared_ptr< _State_base > _S_make_async_state (_BoundFn &&__fn)`
- `template<typename _BoundFn >`
`static std::shared_ptr< _State_base > _S_make_deferred_state (_BoundFn &&__fn)`
- `template<typename _Res_ptr , typename _BoundFn >`
`static _Task_setter< _Res_ptr, _BoundFn > _S_task_setter (_Res_ptr &__ptr, _BoundFn &__call)`

Protected Types

- `typedef __future_base::Result< _Res & > & __result_type`

Protected Member Functions

- `__result_type _M_get_result () const`
- `void _M_swap (__basic_future &__that) noexcept`

Friends

- `template<typename _Fn , typename... _Args>`
`future< __async_result_of< _Fn, _Args... > > async (launch, _Fn &&, _Args &&...)`
- `template<typename >`
`class packaged_task`
- `class promise< _Res & >`

5.725.1 Detailed Description

```
template<typename _Res>
class std::future< _Res & >
```

Partial specialization for `future<R&>`

Definition at line 774 of file `future`.

5.725.2 Member Typedef Documentation

5.725.2.1 `template<typename _Res > using std::future_base::Ptr = unique_ptr< _Res, Result_base::Deleter>`
`[inherited]`

A `unique_ptr` for result objects.

Definition at line 214 of file `future`.

5.725.3 Constructor & Destructor Documentation

5.725.3.1 `template<typename _Res > std::future< _Res >::future (future< _Res > && __uf)` `[inline]`,
`[noexcept]`

Move constructor.

Definition at line 792 of file `future`.

5.725.4 Member Function Documentation

5.725.4.1 `__result_type std::__basic_future< _Res >::M_get_result () const` `[inline]`, `[protected]`,
`[inherited]`

Wait for the state to be ready and rethrow any stored exception.

Definition at line 684 of file `future`.

5.725.4.2 `template<typename _Res > _Res& std::future< _Res >::get ()` `[inline]`

Retrieving the value.

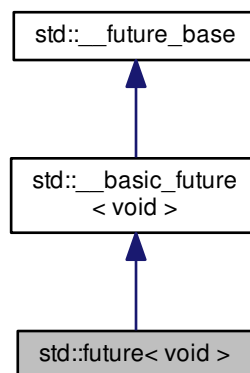
Definition at line 806 of file `future`.

The documentation for this class was generated from the following file:

- [future](#)

5.726 `std::future< void >` Class Template Reference

Inheritance diagram for `std::future< void >`:



Public Types

- `template<typename _Res >`
`using _Ptr = unique_ptr< _Res, _Result_base::_Deleter >`
- `using _State_base = _State_baseV2`

Public Member Functions

- `future (future &&__uf) noexcept`
- `future (const future &)=delete`
- `void get ()`
- `future & operator= (const future &)=delete`
- `future & operator= (future &&__fut) noexcept`
- `shared_future< void > share ()`
- `bool valid () const noexcept`
- `void wait () const`
- `future_status wait_for (const chrono::duration< _Rep, _Period > &__rel) const`
- `future_status wait_until (const chrono::time_point< _Clock, _Duration > &__abs) const`

Static Public Member Functions

- `template<typename _Res , typename _Allocator >`
`static _Ptr< _Result_alloc< _Res, _Allocator > > _S_allocate_result (const _Allocator &__a)`
- `template<typename _Res , typename _Tp >`
`static _Ptr< _Result< _Res > > _S_allocate_result (const std::allocator< _Tp > &__a)`
- `template<typename _BoundFn >`
`static std::shared_ptr< _State_base > _S_make_async_state (_BoundFn &&__fn)`
- `template<typename _BoundFn >`
`static std::shared_ptr< _State_base > _S_make_deferred_state (_BoundFn &&__fn)`
- `template<typename _Res_ptr , typename _BoundFn >`
`static _Task_setter< _Res_ptr, _BoundFn > _S_task_setter (_Res_ptr &__ptr, _BoundFn &__call)`

Protected Types

- `typedef _future_base::_Result< void > & __result_type`

Protected Member Functions

- `__result_type _M_get_result () const`
- `void _M_swap (_basic_future &__that) noexcept`

Friends

- `template<typename _Fn , typename... _Args>`
`future< __async_result_of< _Fn, _Args... > > async (launch, _Fn &&, _Args &&...)`
- `template<typename >`
`class packaged_task`
- `class promise< void >`

5.726.1 Detailed Description

```
template<>
class std::future< void >
```

Explicit specialization for `future<void>`

Definition at line 817 of file `future`.

5.726.2 Member Typedef Documentation

5.726.2.1 `template<typename _Res > using std::__future_base::_Ptr = unique_ptr<_Res, _Result_base::_Deleter>`
[*inherited*]

A `unique_ptr` for result objects.

Definition at line 214 of file `future`.

5.726.3 Constructor & Destructor Documentation

5.726.3.1 `std::future< void >::future (future< void > && __uf)` [*inline*], [*noexcept*]

Move constructor.

Definition at line 835 of file `future`.

5.726.4 Member Function Documentation

5.726.4.1 `__result_type std::__basic_future< void >::_M_get_result () const` [*inline*], [*protected*],
[*inherited*]

Wait for the state to be ready and rethrow any stored exception.

Definition at line 684 of file `future`.

5.726.4.2 `void std::future< void >::get ()` [*inline*]

Retrieving the value.

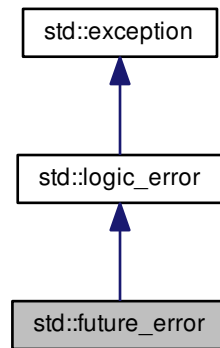
Definition at line 849 of file `future`.

The documentation for this class was generated from the following file:

- [future](#)

5.727 `std::future_error` Class Reference

Inheritance diagram for `std::future_error`:



Public Member Functions

- **future_error** ([error_code](#) __ec)
- const [error_code](#) & **code** () const noexcept
- virtual const char * **what** () const noexcept

5.727.1 Detailed Description

Exception type thrown by futures.

Definition at line 95 of file future.

5.727.2 Member Function Documentation

5.727.2.1 virtual const char* `std::future_error::what () const` [virtual], [noexcept]

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::logic_error](#).

The documentation for this class was generated from the following file:

- [future](#)

5.728 `std::gamma_distribution<_RealType>` Class Template Reference

Classes

- struct [param_type](#)

Public Types

- typedef `_RealType` [result_type](#)

Public Member Functions

- [gamma_distribution](#) (`_RealType` __alpha_val=`_RealType`(1), `_RealType` __beta_val=`_RealType`(1))
- [gamma_distribution](#) (const [param_type](#) &__p)
- template<typename `_ForwardIterator` , typename `_UniformRandomNumberGenerator` >
void [__generate](#) (`_ForwardIterator` __f, `_ForwardIterator` __t, `_UniformRandomNumberGenerator` &__urng)
- template<typename `_ForwardIterator` , typename `_UniformRandomNumberGenerator` >
void [__generate](#) (`_ForwardIterator` __f, `_ForwardIterator` __t, `_UniformRandomNumberGenerator` &__urng, const [param_type](#) &__p)
- template<typename `_UniformRandomNumberGenerator` >
void [__generate](#) ([result_type](#) *__f, [result_type](#) *__t, `_UniformRandomNumberGenerator` &__urng, const [param_type](#) &__p)
- `_RealType` [alpha](#) () const
- `_RealType` [beta](#) () const
- [result_type](#) [max](#) () const
- [result_type](#) [min](#) () const
- template<typename `_UniformRandomNumberGenerator` >
[result_type](#) [operator](#)() (`_UniformRandomNumberGenerator` &__urng)
- template<typename `_UniformRandomNumberGenerator` >
[result_type](#) [operator](#)() (`_UniformRandomNumberGenerator` &__urng, const [param_type](#) &__p)
- [param_type](#) [param](#) () const
- void [param](#) (const [param_type](#) &__param)
- void [reset](#) ()

Friends

- template<typename `_RealType1` , typename `_CharT` , typename `_Traits` >
[std::basic_ostream](#)< `_CharT`, `_Traits` > & [operator<<](#) ([std::basic_ostream](#)< `_CharT`, `_Traits` > &__os, const [std::gamma_distribution](#)< `_RealType1` > &__x)
- bool [operator==](#) (const [gamma_distribution](#) &__d1, const [gamma_distribution](#) &__d2)
- template<typename `_RealType1` , typename `_CharT` , typename `_Traits` >
[std::basic_istream](#)< `_CharT`, `_Traits` > & [operator>>](#) ([std::basic_istream](#)< `_CharT`, `_Traits` > &__is, [std::gamma_distribution](#)< `_RealType1` > &__x)

5.728.1 Detailed Description

```
template<typename _RealType = double>
class std::gamma_distribution< _RealType >
```

A gamma continuous distribution for random numbers.

The formula for the gamma probability density function is:

$$p(x|\alpha, \beta) = \frac{1}{\beta\Gamma(\alpha)} (x/\beta)^{\alpha-1} e^{-x/\beta}$$

Definition at line 2337 of file random.h.

5.728.2 Member Typedef Documentation

5.728.2.1 `template<typename _RealType = double> typedef _RealType std::gamma_distribution< _RealType >::result_type`

The type of the range of the distribution.

Definition at line 2340 of file random.h.

5.728.3 Constructor & Destructor Documentation

5.728.3.1 `template<typename _RealType = double> std::gamma_distribution< _RealType >::gamma_distribution (_RealType __alpha_val = _RealType(1), _RealType __beta_val = _RealType(1)) [inline], [explicit]`

Constructs a gamma distribution with parameters α and β .

Definition at line 2389 of file random.h.

5.728.4 Member Function Documentation

5.728.4.1 `template<typename _RealType = double> _RealType std::gamma_distribution< _RealType >::alpha () const [inline]`

Returns the α of the distribution.

Definition at line 2410 of file random.h.

5.728.4.2 `template<typename _RealType = double> _RealType std::gamma_distribution< _RealType >::beta () const [inline]`

Returns the β of the distribution.

Definition at line 2417 of file random.h.

5.728.4.3 `template<typename _RealType = double> result_type std::gamma_distribution<_RealType>::max () const`
`[inline]`

Returns the least upper bound value of the distribution.

Definition at line 2446 of file random.h.

5.728.4.4 `template<typename _RealType = double> result_type std::gamma_distribution<_RealType>::min () const`
`[inline]`

Returns the greatest lower bound value of the distribution.

Definition at line 2439 of file random.h.

5.728.4.5 `template<typename _RealType = double> template<typename _UniformRandomNumberGenerator> result_type`
`std::gamma_distribution<_RealType>::operator() (_UniformRandomNumberGenerator &__urng) [inline]`

Generating functions.

Definition at line 2454 of file random.h.

Referenced by std::operator>>().

5.728.4.6 `template<typename _RealType> template<typename _UniformRandomNumberGenerator>`
`gamma_distribution<_RealType>::result_type std::gamma_distribution<_RealType>::operator() (`
`_UniformRandomNumberGenerator &__urng, const param_type &__param)`

Marsaglia, G. and Tsang, W. W. "A Simple Method for Generating Gamma Variables" ACM Transactions on Mathematical Software, 26, 3, 363-372, 2000.

Definition at line 2337 of file bits/random.tcc.

References std::dec(), std::basic_ios<_CharT, _Traits>::fill(), std::ios_base::flags(), std::left(), std::log(), std::weibull_distribution<_RealType>::operator(), std::__detail::operator>>(), std::gamma_distribution<_RealType>::param(), std::pow(), std::ios_base::precision(), std::scientific(), std::skipws(), and std::basic_ios<_CharT, _Traits>::widen().

5.728.4.7 `template<typename _RealType = double> param_type std::gamma_distribution<_RealType>::param () const`
`[inline]`

Returns the parameter set of the distribution.

Definition at line 2424 of file random.h.

Referenced by std::gamma_distribution<_RealType>::operator()().

5.728.4.8 `template<typename _RealType = double> void std::gamma_distribution<_RealType>::param (const`
`param_type &__param) [inline]`

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

Definition at line 2432 of file random.h.

5.728.4.9 `template<typename _RealType = double> void std::gamma_distribution<_RealType>::reset() [inline]`

Resets the distribution state.

Definition at line 2403 of file random.h.

5.728.5 Friends And Related Function Documentation

5.728.5.1 `template<typename _RealType = double> template<typename _RealType1 , typename _CharT , typename _Traits > std::basic_ostream<_CharT, _Traits>& operator<< (std::basic_ostream<_CharT, _Traits> & __os, const std::gamma_distribution<_RealType1> & __x) [friend]`

Inserts a gamma_distribution random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A gamma_distribution random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

5.728.5.2 `template<typename _RealType = double> bool operator==(const gamma_distribution<_RealType> & __d1, const gamma_distribution<_RealType> & __d2) [friend]`

Return true if two gamma distributions have the same parameters and the sequences that would be generated are equal.

Definition at line 2490 of file random.h.

5.728.5.3 `template<typename _RealType = double> template<typename _RealType1 , typename _CharT , typename _Traits > std::basic_istream<_CharT, _Traits>& operator>> (std::basic_istream<_CharT, _Traits> & __is, std::gamma_distribution<_RealType1> & __x) [friend]`

Extracts a gamma_distribution random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>gamma_distribution</code> random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.729 `std::gamma_distribution<_RealType>::param_type` Struct Reference

Public Types

- typedef `gamma_distribution<_RealType>` **distribution_type**

Public Member Functions

- **param_type** (`_RealType __alpha_val=_RealType(1), _RealType __beta_val=_RealType(1)`)
- `_RealType alpha` () const
- `_RealType beta` () const

Friends

- class **gamma_distribution<_RealType>**
- bool **operator==** (const `param_type` &__p1, const `param_type` &__p2)

5.729.1 Detailed Description

```
template<typename _RealType = double>
struct std::gamma_distribution<_RealType>::param_type
```

Parameter type.

Definition at line 2346 of file `random.h`.

The documentation for this struct was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.730 `std::geometric_distribution<_IntType>` Class Template Reference

Classes

- struct [param_type](#)

Public Types

- typedef `_IntType` [result_type](#)

Public Member Functions

- **`geometric_distribution`** (`double __p=0.5`)
- **`geometric_distribution`** (`const param_type &__p`)
- `template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >`
`void __generate (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)`
- `template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >`
`void __generate (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const param_type &__p)`
- `template<typename _UniformRandomNumberGenerator >`
`void __generate (result_type *__f, result_type *__t, _UniformRandomNumberGenerator &__urng, const param_type &__p)`
- [result_type](#) `max () const`
- [result_type](#) `min () const`
- `template<typename _UniformRandomNumberGenerator >`
`result_type operator() (_UniformRandomNumberGenerator &__urng)`
- `template<typename _UniformRandomNumberGenerator >`
`result_type operator() (_UniformRandomNumberGenerator &__urng, const param_type &__p)`
- `double p () const`
- [param_type](#) `param () const`
- `void param (const param_type &__param)`
- `void reset ()`

Friends

- `bool operator== (const geometric_distribution &__d1, const geometric_distribution &__d2)`

5.730.1 Detailed Description

```
template<typename _IntType = int>
class std::geometric_distribution<_IntType>
```

A discrete geometric random number distribution.

The formula for the geometric probability density function is $p(i|p) = p(1 - p)^i$ where p is the parameter of the distribution.

Definition at line 3843 of file `random.h`.

5.730.2 Member Typedef Documentation

5.730.2.1 `template<typename _IntType = int> typedef _IntType std::geometric_distribution<_IntType>::result_type`

The type of the range of the distribution.

Definition at line 3846 of file `random.h`.

5.730.3 Member Function Documentation

5.730.3.1 `template<typename _IntType = int> result_type std::geometric_distribution<_IntType>::max () const` `[inline]`

Returns the least upper bound value of the distribution.

Definition at line 3935 of file `random.h`.

References `std::numeric_limits<_Tp>::max()`.

5.730.3.2 `template<typename _IntType = int> result_type std::geometric_distribution<_IntType>::min () const` `[inline]`

Returns the greatest lower bound value of the distribution.

Definition at line 3928 of file `random.h`.

5.730.3.3 `template<typename _IntType = int> template<typename _UniformRandomNumberGenerator> result_type` `std::geometric_distribution<_IntType>::operator() (_UniformRandomNumberGenerator & __urng)` `[inline]`

Generating functions.

Definition at line 3943 of file `random.h`.

Referenced by `std::operator<<()`.

5.730.3.4 `template<typename _IntType = int> double std::geometric_distribution<_IntType>::p () const` `[inline]`

Returns the distribution parameter `p`.

Definition at line 3906 of file `random.h`.

5.730.3.5 `template<typename _IntType = int> param_type std::geometric_distribution<_IntType>::param () const` `[inline]`

Returns the parameter set of the distribution.

Definition at line 3913 of file `random.h`.

Referenced by `std::operator>>()`.

5.730.3.6 `template<typename _IntType = int> void std::geometric_distribution<_IntType>::param (const param_type &` `__param)` `[inline]`

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

Definition at line 3921 of file random.h.

5.730.3.7 `template<typename _IntType = int> void std::geometric_distribution<_IntType>::reset () [inline]`

Resets the distribution state.

Does nothing for the geometric distribution.

Definition at line 3900 of file random.h.

5.730.4 Friends And Related Function Documentation

5.730.4.1 `template<typename _IntType = int> bool operator== (const geometric_distribution<_IntType> &__d1, const geometric_distribution<_IntType> &__d2) [friend]`

Return true if two geometric distributions have the same parameters.

Definition at line 3978 of file random.h.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.731 `std::geometric_distribution<_IntType>::param_type` Struct Reference

Public Types

- typedef `geometric_distribution<_IntType>` **distribution_type**

Public Member Functions

- **param_type** (double __p=0.5)
- double **p** () const

Friends

- class **geometric_distribution<_IntType>**
- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)

5.731.1 Detailed Description

```
template<typename _IntType = int>
struct std::geometric_distribution< _IntType >::param_type
```

Parameter type.

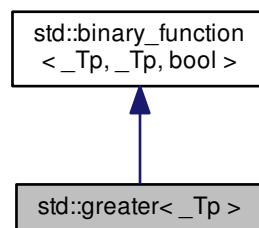
Definition at line 3852 of file random.h.

The documentation for this struct was generated from the following file:

- [random.h](#)

5.732 std::greater< _Tp > Struct Template Reference

Inheritance diagram for std::greater< _Tp >:



Public Types

- typedef _Tp [first_argument_type](#)
- typedef bool [result_type](#)
- typedef _Tp [second_argument_type](#)

Public Member Functions

- _GLIBCXX14_CONSTEXPR bool **operator()** (const _Tp &__x, const _Tp &__y) const

5.732.1 Detailed Description

```
template<typename _Tp>
struct std::greater< _Tp >
```

One of the [comparison functors](#).

Definition at line 337 of file stl_function.h.

5.732.2 Member Typedef Documentation

5.732.2.1 `typedef _Tp std::binary_function<_Tp, _Tp, bool>::first_argument_type` [inherited]

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.732.2.2 `typedef bool std::binary_function<_Tp, _Tp, bool>::result_type` [inherited]

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.732.2.3 `typedef _Tp std::binary_function<_Tp, _Tp, bool>::second_argument_type` [inherited]

`second_argument_type` is the type of the second argument

Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.733 `std::greater< void >` Struct Template Reference

Public Types

- `typedef __is_transparent is_transparent`

Public Member Functions

- `template<typename _Tp, typename _Up> _GLIBCXX14_CONSTEXPR auto operator() (_Tp &&__t, _Up &&__u) const noexcept(noexcept(std::forward<_Tp>\(__t\) > std::forward<_Up>\\(__u\\))) -> decltype(std::forward<_Tp>\(__t\) > std::forward<_Up>\\(__u\\))`

5.733.1 Detailed Description

```
template<>
struct std::greater< void >
```

One of the [comparison functors](#).

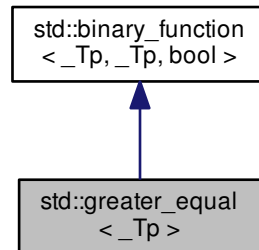
Definition at line 442 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.734 std::greater_equal< _Tp > Struct Template Reference

Inheritance diagram for std::greater_equal< _Tp >:



Public Types

- typedef `_Tp` `first_argument_type`
- typedef `bool` `result_type`
- typedef `_Tp` `second_argument_type`

Public Member Functions

- `_GLIBCXX14_CONSTEXPR bool operator() (const _Tp &__x, const _Tp &__y) const`

5.734.1 Detailed Description

```
template<typename _Tp>
struct std::greater_equal< _Tp >
```

One of the [comparison functors](#).

Definition at line 343 of file `stl_function.h`.

5.734.2 Member Typedef Documentation

5.734.2.1 typedef `_Tp` `std::binary_function< _Tp, _Tp, bool >::first_argument_type` [inherited]

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.734.2.2 `typedef bool std::binary_function<_Tp, _Tp, bool>::result_type` [inherited]

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.734.2.3 `typedef _Tp std::binary_function<_Tp, _Tp, bool>::second_argument_type` [inherited]

`second_argument_type` is the type of the second argument

Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.735 `std::greater_equal< void >` Struct Template Reference

Public Types

- `typedef __is_transparent is_transparent`

Public Member Functions

- `template<typename _Tp, typename _Up >
_GLIBCXX14_CONSTEXPR auto operator() (_Tp &&__t, _Up &&__u) const noexcept(noexcept(std::forward<_Tp>(__t) >= std::forward<_Up>(__u))) -> decltype(std::forward<_Tp>(__t) >= std::forward<_Up>(__u))`

5.735.1 Detailed Description

```
template<>
struct std::greater_equal< void >
```

One of the [comparison functors](#).

Definition at line 472 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.736 std::gslice Class Reference

Public Member Functions

- [gslice](#) ()
- [gslice](#) (size_t __o, const [valarray](#)< size_t > &__l, const [valarray](#)< size_t > &__s)
- [gslice](#) (const [gslice](#) &)
- [~gslice](#) ()
- [gslice](#) & [operator=](#) (const [gslice](#) &)
- [valarray](#)< size_t > [size](#) () const
- size_t [start](#) () const
- [valarray](#)< size_t > [stride](#) () const

Friends

- template<typename _Tp >
class [valarray](#)

5.736.1 Detailed Description

Class defining multi-dimensional subset of an array.

The slice class represents a multi-dimensional subset of an array, specified by three parameter sets: start offset, size array, and stride array. The start offset is the index of the first element of the array that is part of the subset. The size and stride array describe each dimension of the slice. Size is the number of elements in that dimension, and stride is the distance in the array between successive elements in that dimension. Each dimension's size and stride is taken to begin at an array element described by the previous dimension. The size array and stride array must be the same size.

For example, if you have offset==3, stride[0]==11, size[1]==3, stride[1]==3, then slice[0,0]==array[3], slice[0,1]==array[6], slice[0,2]==array[9], slice[1,0]==array[14], slice[1,1]==array[17], slice[1,2]==array[20].

Definition at line 64 of file gslice.h.

The documentation for this class was generated from the following file:

- [gslice.h](#)

5.737 std::gslice_array< _Tp > Class Template Reference

Public Types

- typedef _Tp [value_type](#)

Public Member Functions

- [gslice_array](#) (const [gslice_array](#) &)
- void [operator%=>](#) (const [valarray](#)<_Tp> &) const
- template<class _Dom >
void [operator%=>](#) (const _Expr<_Dom, _Tp> &) const
- void [operator&=>](#) (const [valarray](#)<_Tp> &) const
- template<class _Dom >
void [operator&=>](#) (const _Expr<_Dom, _Tp> &) const
- void [operator*=>](#) (const [valarray](#)<_Tp> &) const
- template<class _Dom >
void [operator*=>](#) (const _Expr<_Dom, _Tp> &) const
- void [operator+=>](#) (const [valarray](#)<_Tp> &) const
- template<class _Dom >
void [operator+=>](#) (const _Expr<_Dom, _Tp> &) const
- void [operator-=>](#) (const [valarray](#)<_Tp> &) const
- template<class _Dom >
void [operator-=>](#) (const _Expr<_Dom, _Tp> &) const
- void [operator/=>](#) (const [valarray](#)<_Tp> &) const
- template<class _Dom >
void [operator/=>](#) (const _Expr<_Dom, _Tp> &) const
- void [operator<=>](#) (const [valarray](#)<_Tp> &) const
- template<class _Dom >
void [operator<=>](#) (const _Expr<_Dom, _Tp> &) const
- [gslice_array](#) & [operator=](#) (const [gslice_array](#) &)
- void [operator=](#) (const [valarray](#)<_Tp> &) const
- void [operator=](#) (const _Tp &) const
- template<class _Dom >
void [operator=](#) (const _Expr<_Dom, _Tp> &) const
- void [operator>=>](#) (const [valarray](#)<_Tp> &) const
- template<class _Dom >
void [operator>=>](#) (const _Expr<_Dom, _Tp> &) const
- void [operator^=>](#) (const [valarray](#)<_Tp> &) const
- template<class _Dom >
void [operator^=>](#) (const _Expr<_Dom, _Tp> &) const
- void [operator|=>](#) (const [valarray](#)<_Tp> &) const
- template<class _Dom >
void [operator|=>](#) (const _Expr<_Dom, _Tp> &) const

Friends

- class [valarray](#)<_Tp>

5.737.1 Detailed Description

```
template<typename _Tp>
class std::gslice_array<_Tp>
```

Reference to multi-dimensional subset of an array.

A [gslice_array](#) is a reference to the actual elements of an array specified by a [gslice](#). The way to get a [gslice_array](#) is to call [operator\[\]](#)([gslice](#)) on a [valarray](#). The returned [gslice_array](#) then permits carrying operations out on the referenced subset of elements in the original [valarray](#). For example, [operator+=\(valarray\)](#) will add values to the subset of elements in the underlying [valarray](#) this [gslice_array](#) refers to.

Parameters

<i>Tp</i>	Element type.
-----------	---------------

Definition at line 82 of file `valarray`.

The documentation for this class was generated from the following files:

- [valarray](#)
- [gslice_array.h](#)

5.738 `std::hash<_Tp>` Struct Template Reference

Inherits `std::__hash_enum<_Tp, bool>`.

5.738.1 Detailed Description

```
template<typename _Tp>
struct std::hash<_Tp>
```

Primary class template hash.

Primary class template hash, usable for enum types only.

Definition at line 134 of file `system_error`.

The documentation for this struct was generated from the following file:

- [system_error](#)

5.739 `std::hash<__debug::bitset<_Nb>>` Struct Template Reference

Inherits `std::__hash_base<_Result, _Arg>`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t` **operator()** (const [__debug::bitset](#)<_Nb> &__b) const noexcept

5.739.1 Detailed Description

```
template<size_t _Nb>
struct std::hash<__debug::bitset<_Nb>>
```

std::hash specialization for bitset.

Definition at line 415 of file debug/bitset.

The documentation for this struct was generated from the following file:

- [debug/bitset](#)

5.740 std::hash<__debug::vector< bool, _Alloc >> Struct Template Reference

Inherits std::__hash_base<_Result, _Arg>.

Public Types

- typedef _Arg **argument_type**
- typedef _Result **result_type**

Public Member Functions

- size_t **operator()** (const [__debug::vector](#)< bool, _Alloc > &__b) const noexcept

5.740.1 Detailed Description

```
template<typename _Alloc>
struct std::hash<__debug::vector< bool, _Alloc >>
```

std::hash specialization for vector<bool>.

Definition at line 757 of file debug/vector.

The documentation for this struct was generated from the following file:

- [debug/vector](#)

5.741 std::hash<__gnu_cxx::__u16vstring> Struct Template Reference

Inherits std::__hash_base<_Result, _Arg>.

Public Types

- typedef _Arg **argument_type**
- typedef _Result **result_type**

Public Member Functions

- size_t **operator()** (const [__gnu_cxx::__u16vstring](#) &__s) const noexcept

5.741.1 Detailed Description

```
template<>
struct std::hash< __gnu_cxx::__u16vstring >
```

std::hash specialization for __u16vstring.

Definition at line 2939 of file vstring.h.

The documentation for this struct was generated from the following file:

- [vstring.h](#)

5.742 std::hash< __gnu_cxx::__u32vstring > Struct Template Reference

Inherits std::__hash_base< _Result, _Arg >.

Public Types

- typedef _Arg **argument_type**
- typedef _Result **result_type**

Public Member Functions

- size_t **operator()** (const [__gnu_cxx::__u32vstring](#) &__s) const noexcept

5.742.1 Detailed Description

```
template<>
struct std::hash< __gnu_cxx::__u32vstring >
```

std::hash specialization for __u32vstring.

Definition at line 2950 of file vstring.h.

The documentation for this struct was generated from the following file:

- [vstring.h](#)

5.743 `std::hash< __gnu_cxx::__vstring >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (const [__gnu_cxx::__vstring](#) &__s) const noexcept

5.743.1 Detailed Description

```
template<>
struct std::hash< __gnu_cxx::__vstring >
```

`std::hash` specialization for `__vstring`.

Definition at line 2915 of file `vstring.h`.

The documentation for this struct was generated from the following file:

- [vstring.h](#)

5.744 `std::hash< __gnu_cxx::__wvstring >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (const [__gnu_cxx::__wvstring](#) &__s) const noexcept

5.744.1 Detailed Description

```
template<>
struct std::hash< __gnu_cxx::__wvstring >
```

std::hash specialization for __wvstring.

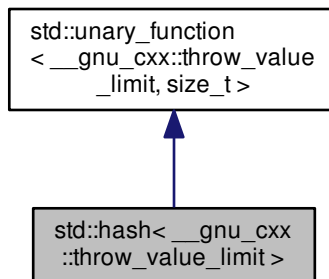
Definition at line 2926 of file vstring.h.

The documentation for this struct was generated from the following file:

- [vstring.h](#)

5.745 std::hash< __gnu_cxx::throw_value_limit > Struct Template Reference

Inheritance diagram for std::hash< __gnu_cxx::throw_value_limit >:



Public Types

- typedef `__gnu_cxx::throw_value_limit` `argument_type`
- typedef `size_t` `result_type`

Public Member Functions

- `size_t operator()` (const `__gnu_cxx::throw_value_limit` &`__val`) const

5.745.1 Detailed Description

```
template<>
struct std::hash< __gnu_cxx::throw_value_limit >
```

Explicit specialization of std::hash for __gnu_cxx::throw_value_limit.

Definition at line 950 of file throw_allocator.h.

5.745.2 Member Typedef Documentation

5.745.2.1 `typedef __gnu_cxx::throw_value_limit std::unary_function< __gnu_cxx::throw_value_limit , size_t >::argument_type` [inherited]

argument_type is the type of the argument

Definition at line 108 of file stl_function.h.

5.745.2.2 `typedef size_t std::unary_function< __gnu_cxx::throw_value_limit , size_t >::result_type` [inherited]

result_type is the return type

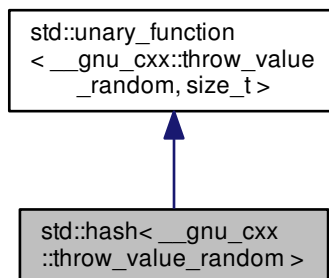
Definition at line 111 of file stl_function.h.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

5.746 std::hash< __gnu_cxx::throw_value_random > Struct Template Reference

Inheritance diagram for std::hash< __gnu_cxx::throw_value_random >:



Public Types

- typedef `__gnu_cxx::throw_value_random_argument_type`
- typedef `size_t` `result_type`

Public Member Functions

- `size_t operator()` (const `__gnu_cxx::throw_value_random` &__val) const

5.746.1 Detailed Description

template<>
struct `std::hash<__gnu_cxx::throw_value_random>`

Explicit specialization of `std::hash` for `__gnu_cxx::throw_value_random`.

Definition at line 965 of file `throw_allocator.h`.

5.746.2 Member Typedef Documentation

5.746.2.1 typedef `__gnu_cxx::throw_value_random` `std::unary_function<__gnu_cxx::throw_value_random, size_t>::argument_type` [inherited]

`argument_type` is the type of the argument

Definition at line 108 of file `stl_function.h`.

5.746.2.2 typedef `size_t` `std::unary_function<__gnu_cxx::throw_value_random, size_t>::result_type` [inherited]

`result_type` is the return type

Definition at line 111 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [throw_allocator.h](#)

5.747 `std::hash<__profile::bitset<_Nb>>` Struct Template Reference

Inherits `std::__hash_base<_Result, _Arg>`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (const [__profile::bitset](#)< `_Nb` > &__b) const noexcept

5.747.1 Detailed Description

```
template<size_t _Nb>
struct std::hash< __profile::bitset< _Nb > >
```

std::hash specialization for `bitset`.

Definition at line 234 of file `profile/bitset`.

The documentation for this struct was generated from the following file:

- [profile/bitset](#)

5.748 std::hash< __profile::vector< bool, _Alloc > > Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (const `__profile::vector`< `bool`, `_Alloc` > &__b) const noexcept

5.748.1 Detailed Description

```
template<typename _Alloc>
struct std::hash< __profile::vector< bool, _Alloc > >
```

std::hash specialization for `vector`<`bool`>.

Definition at line 559 of file `profile/vector`.

The documentation for this struct was generated from the following file:

- [profile/vector](#)

5.749 `std::hash< __shared_ptr< _Tp, _Lp > >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`const __shared_ptr< _Tp, _Lp > &__s`) `const` `noexcept`

5.749.1 Detailed Description

```
template<typename _Tp, _Lock_policy _Lp>
struct std::hash< __shared_ptr< _Tp, _Lp > >
```

`std::hash` specialization for `__shared_ptr`.

Definition at line 1588 of file `shared_ptr_base.h`.

The documentation for this struct was generated from the following file:

- [shared_ptr_base.h](#)

5.750 `std::hash< _Tp * >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`_Tp *__p`) `const` `noexcept`

5.750.1 Detailed Description

```
template<typename _Tp>
struct std::hash< _Tp * >
```

Partial specializations for pointer types.

Definition at line 90 of file functional_hash.h.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

5.751 std::hash< bool > Struct Template Reference

Inherits std::__hash_base< _Result, _Arg >.

Public Types

- typedef _Arg **argument_type**
- typedef _Result **result_type**

Public Member Functions

- size_t **operator()** (bool __val) const noexcept

5.751.1 Detailed Description

```
template<>
struct std::hash< bool >
```

Explicit specialization for bool.

Definition at line 108 of file functional_hash.h.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

5.752 std::hash< char > Struct Template Reference

Inherits std::__hash_base< _Result, _Arg >.

Public Types

- typedef _Arg **argument_type**
- typedef _Result **result_type**

Public Member Functions

- size_t **operator()** (char __val) const noexcept

5.752.1 Detailed Description

```
template<>
struct std::hash< char >
```

Explicit specialization for char.

Definition at line 111 of file functional_hash.h.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

5.753 std::hash< char16_t > Struct Template Reference

Inherits std::__hash_base< _Result, _Arg >.

Public Types

- typedef _Arg **argument_type**
- typedef _Result **result_type**

Public Member Functions

- size_t **operator()** (char16_t __val) const noexcept

5.753.1 Detailed Description

```
template<>
struct std::hash< char16_t >
```

Explicit specialization for char16_t.

Definition at line 123 of file functional_hash.h.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

5.754 `std::hash< char32_t >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`char32_t __val`) `const noexcept`

5.754.1 Detailed Description

```
template<>
struct std::hash< char32_t >
```

Explicit specialization for `char32_t`.

Definition at line 126 of file `functional_hash.h`.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

5.755 `std::hash< double >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`double __val`) `const noexcept`

5.755.1 Detailed Description

```
template<>
struct std::hash< double >
```

Specialization for double.

Definition at line 221 of file `functional_hash.h`.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

5.756 `std::hash< error_code >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t` **operator()** (const [error_code](#) &__e) const noexcept

5.756.1 Detailed Description

```
template<>
struct std::hash< error_code >
```

`std::hash` specialization for `error_code`.

Definition at line 379 of file `system_error`.

The documentation for this struct was generated from the following file:

- [system_error](#)

5.757 `std::hash< experimental::shared_ptr< _Tp > >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (const `experimental::shared_ptr<_Tp>` &__s) const noexcept

5.757.1 Detailed Description

```
template<typename _Tp>
struct std::hash< experimental::shared_ptr<_Tp> > >
```

`std::hash` specialization for `shared_ptr`.

Definition at line 1317 of file `experimental/bits/shared_ptr.h`.

The documentation for this struct was generated from the following file:

- [experimental/bits/shared_ptr.h](#)

5.758 `std::hash< float >` Struct Template Reference

Inherits `std::__hash_base<_Result, _Arg>`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (float __val) const noexcept

5.758.1 Detailed Description

```
template<>
struct std::hash< float >
```

Specialization for `float`.

Definition at line 209 of file `functional_hash.h`.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

5.759 `std::hash< int >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t` **operator()** (`int __val`) const noexcept

5.759.1 Detailed Description

```
template<>
struct std::hash< int >
```

Explicit specialization for `int`.

Definition at line 132 of file `functional_hash.h`.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

5.760 `std::hash< long >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t` **operator()** (`long __val`) const noexcept

5.760.1 Detailed Description

```
template<>
struct std::hash< long >
```

Explicit specialization for long.

Definition at line 135 of file functional_hash.h.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

5.761 std::hash< long double > Struct Template Reference

Inherits std::__hash_base< _Result, _Arg >.

Public Types

- typedef _Arg **argument_type**
- typedef _Result **result_type**

Public Member Functions

- size_t **operator()** (long double __val) const noexcept

5.761.1 Detailed Description

```
template<>
struct std::hash< long double >
```

Specialization for long double.

Definition at line 233 of file functional_hash.h.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

5.762 std::hash< long long > Struct Template Reference

Inherits std::__hash_base< _Result, _Arg >.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (long long __val) const noexcept

5.762.1 Detailed Description

```
template<>
struct std::hash< long long >
```

Explicit specialization for long long.

Definition at line 138 of file `functional_hash.h`.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

5.763 `std::hash< shared_ptr< _Tp > >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (const [shared_ptr< _Tp >](#) &__s) const noexcept

5.763.1 Detailed Description

```
template<typename _Tp>
struct std::hash< shared_ptr< _Tp > >
```

`std::hash` specialization for `shared_ptr`.

Definition at line 641 of file `bits/shared_ptr.h`.

The documentation for this struct was generated from the following file:

- [bits/shared_ptr.h](#)

5.764 `std::hash< short >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t` **operator()** (`short __val`) `const noexcept`

5.764.1 Detailed Description

```
template<>
struct std::hash< short >
```

Explicit specialization for `short`.

Definition at line 129 of file `functional_hash.h`.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

5.765 `std::hash< signed char >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t` **operator()** (`signed char __val`) `const noexcept`

5.765.1 Detailed Description

```
template<>
struct std::hash< signed char >
```

Explicit specialization for signed char.

Definition at line 114 of file functional_hash.h.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

5.766 std::hash< string > Struct Template Reference

Inherits std::__hash_base< _Result, _Arg >.

Public Types

- typedef _Arg **argument_type**
- typedef _Result **result_type**

Public Member Functions

- size_t **operator()** (const [string](#) &__s) const noexcept

5.766.1 Detailed Description

```
template<>
struct std::hash< string >
```

std::hash specialization for string.

Definition at line 5639 of file basic_string.h.

The documentation for this struct was generated from the following file:

- [basic_string.h](#)

5.767 std::hash< thread::id > Struct Template Reference

Inherits std::__hash_base< _Result, _Arg >.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (const [thread::id](#) &__id) const noexcept

5.767.1 Detailed Description

```
template<>
struct std::hash< thread::id >
```

std::hash specialization for thread::id.

Definition at line 252 of file thread.

The documentation for this struct was generated from the following file:

- [thread](#)

5.768 std::hash< type_index > Struct Template Reference

Public Types

- typedef [type_index](#) **argument_type**
- typedef `size_t` **result_type**

Public Member Functions

- `size_t operator()` (const [type_index](#) &__ti) const noexcept

5.768.1 Detailed Description

```
template<>
struct std::hash< type_index >
```

std::hash specialization for type_index.

Definition at line 97 of file typeindex.

The documentation for this struct was generated from the following file:

- [typeindex](#)

5.769 `std::hash< u16string >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (const `u16string` &__s) const noexcept

5.769.1 Detailed Description

```
template<>
struct std::hash< u16string >
```

`std::hash` specialization for `u16string`.

Definition at line 5672 of file `basic_string.h`.

The documentation for this struct was generated from the following file:

- [basic_string.h](#)

5.770 `std::hash< u32string >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (const `u32string` &__s) const noexcept

5.770.1 Detailed Description

```
template<>
struct std::hash< u32string >
```

std::hash specialization for u32string.

Definition at line 5687 of file basic_string.h.

The documentation for this struct was generated from the following file:

- [basic_string.h](#)

5.771 std::hash< unique_ptr< _Tp, _Dp > > Struct Template Reference

Inherits std::__hash_base< _Result, _Arg >.

Public Types

- typedef _Arg **argument_type**
- typedef _Result **result_type**

Public Member Functions

- size_t **operator() (**const [unique_ptr](#)< _Tp, _Dp > &__u) const noexcept

5.771.1 Detailed Description

```
template<typename _Tp, typename _Dp>
struct std::hash< unique_ptr< _Tp, _Dp > >
```

std::hash specialization for unique_ptr.

Definition at line 760 of file unique_ptr.h.

The documentation for this struct was generated from the following file:

- [unique_ptr.h](#)

5.772 std::hash< unsigned char > Struct Template Reference

Inherits std::__hash_base< _Result, _Arg >.

Public Types

- typedef _Arg **argument_type**
- typedef _Result **result_type**

Public Member Functions

- size_t **operator()** (unsigned char __val) const noexcept

5.772.1 Detailed Description

template<>
struct std::hash< unsigned char >

Explicit specialization for unsigned char.

Definition at line 117 of file functional_hash.h.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

5.773 std::hash< unsigned int > Struct Template Reference

Inherits std::__hash_base< _Result, _Arg >.

Public Types

- typedef _Arg **argument_type**
- typedef _Result **result_type**

Public Member Functions

- size_t **operator()** (unsigned int __val) const noexcept

5.773.1 Detailed Description

template<>
struct std::hash< unsigned int >

Explicit specialization for unsigned int.

Definition at line 144 of file functional_hash.h.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

5.774 `std::hash< unsigned long >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t` **operator()** (`unsigned long __val`) `const noexcept`

5.774.1 Detailed Description

```
template<>
struct std::hash< unsigned long >
```

Explicit specialization for unsigned long.

Definition at line 147 of file `functional_hash.h`.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

5.775 `std::hash< unsigned long long >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t` **operator()** (`unsigned long long __val`) `const noexcept`

5.775.1 Detailed Description

```
template<>
struct std::hash< unsigned long long >
```

Explicit specialization for unsigned long long.

Definition at line 150 of file `functional_hash.h`.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

5.776 `std::hash< unsigned short >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t` **operator()** (`unsigned short __val`) `const noexcept`

5.776.1 Detailed Description

```
template<>
struct std::hash< unsigned short >
```

Explicit specialization for unsigned short.

Definition at line 141 of file `functional_hash.h`.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

5.777 `std::hash< wchar_t >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`wchar_t __val`) `const noexcept`

5.777.1 Detailed Description

```
template<>
struct std::hash< wchar_t >
```

Explicit specialization for `wchar_t`.

Definition at line 120 of file `functional_hash.h`.

The documentation for this struct was generated from the following file:

- [functional_hash.h](#)

5.778 `std::hash< wstring >` Struct Template Reference

Inherits `std::__hash_base< _Result, _Arg >`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`const wstring &__s`) `const noexcept`

5.778.1 Detailed Description

```
template<>
struct std::hash< wstring >
```

`std::hash` specialization for `wstring`.

Definition at line 5654 of file `basic_string.h`.

The documentation for this struct was generated from the following file:

- [basic_string.h](#)

5.779 `std::hash<::bitset<_Nb>>` Struct Template Reference

Inherits `std::__hash_base<_Result, _Arg>`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`const ::bitset<_Nb> &__b`) `const noexcept`

5.779.1 Detailed Description

```
template<size_t _Nb>
struct std::hash<::bitset<_Nb>>
```

`std::hash` specialization for `bitset`.

Definition at line 1561 of file `bitset`.

The documentation for this struct was generated from the following file:

- [bitset](#)

5.780 `std::hash<::vector<bool, _Alloc>>` Struct Template Reference

Inherits `std::__hash_base<_Result, _Arg>`.

Public Types

- typedef `_Arg` **argument_type**
- typedef `_Result` **result_type**

Public Member Functions

- `size_t operator()` (`const ::vector<bool, _Alloc> &`) `const noexcept`

5.780.1 Detailed Description

```
template<typename _Alloc>
struct std::hash<::vector< bool, _Alloc > >
```

std::hash specialization for vector<bool>.

Definition at line 1271 of file stl_bvector.h.

The documentation for this struct was generated from the following files:

- [stl_bvector.h](#)
- [vector.tcc](#)

5.781 std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > Class Template Reference

Public Types

- typedef _UIntType [result_type](#)

Public Member Functions

- [independent_bits_engine](#) ()
- [independent_bits_engine](#) (const _RandomNumberEngine &__rng)
- [independent_bits_engine](#) (_RandomNumberEngine &&__rng)
- [independent_bits_engine](#) ([result_type](#) __s)
- template<typename _Sseq, typename = typename std::enable_if<!std::is_same<_Sseq, independent_bits_engine>::value && !std::is_↵
same<_Sseq, _RandomNumberEngine>::value> ::type>
[independent_bits_engine](#) (_Sseq &__q)
- const _RandomNumberEngine & [base](#) () const noexcept
- void [discard](#) (unsigned long long __z)
- [result_type](#) [operator](#)() ()
- void [seed](#) ()
- void [seed](#) ([result_type](#) __s)
- template<typename _Sseq >
void [seed](#) (_Sseq &__q)

Static Public Member Functions

- static constexpr [result_type](#) [max](#) ()
- static constexpr [result_type](#) [min](#) ()

Friends

- bool [operator==](#) (const [independent_bits_engine](#) &__lhs, const [independent_bits_engine](#) &__rhs)
- template<typename _CharT, typename _Traits >
[std::basic_istream](#)< _CharT, _Traits > & [operator>>](#) ([std::basic_istream](#)< _CharT, _Traits > &__is, [std::↵
::independent_bits_engine](#)< _RandomNumberEngine, __w, _UIntType > &__x)

5.781.1 Detailed Description

```
template<typename _RandomNumberEngine, size_t __w, typename _UIntType>
class std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType>
```

Produces random numbers by combining random numbers from some base engine to produce random numbers with a specifies number of bits `__w`.

Definition at line 1066 of file `random.h`.

5.781.2 Member Typedef Documentation

5.781.2.1 `template<typename _RandomNumberEngine, size_t __w, typename _UIntType> typedef _UIntType std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType>::result_type`

The type of the generated random value.

Definition at line 1069 of file `random.h`.

5.781.3 Constructor & Destructor Documentation

5.781.3.1 `template<typename _RandomNumberEngine, size_t __w, typename _UIntType> std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType>::independent_bits_engine() [inline]`

Constructs a default `independent_bits_engine` engine.

The underlying engine is default constructed as well.

Definition at line 1082 of file `random.h`.

5.781.3.2 `template<typename _RandomNumberEngine, size_t __w, typename _UIntType> std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType>::independent_bits_engine(const _RandomNumberEngine & __rng) [inline], [explicit]`

Copy constructs a `independent_bits_engine` engine.

Copies an existing base class random number generator.

Parameters

<code>__rng</code>	An existing (base class) engine object.
--------------------	---

Definition at line 1092 of file `random.h`.


```
5.781.3.3  template<typename _RandomNumberEngine, size_t __w, typename _UIntType> std::independent_bits_engine<
            _RandomNumberEngine, __w, _UIntType >::independent_bits_engine ( _RandomNumberEngine && __rng )
            [inline], [explicit]
```

Move constructs a independent_bits_engine engine.

Copies an existing base class random number generator.

Parameters

<code>__rng</code>	An existing (base class) engine object.
--------------------	---

Definition at line 1102 of file random.h.

```
5.781.3.4  template<typename _RandomNumberEngine, size_t __w, typename _UIntType> std::independent_bits_engine<
            _RandomNumberEngine, __w, _UIntType >::independent_bits_engine ( result_type __s ) [inline],
            [explicit]
```

Seed constructs a independent_bits_engine engine.

Constructs the underlying generator engine seeded with __s.

Parameters

<code>__s</code>	A seed value for the base class engine.
------------------	---

Definition at line 1112 of file random.h.

```
5.781.3.5  template<typename _RandomNumberEngine, size_t __w, typename _UIntType> template<typename _Sseq, typename
            = typename std::enable_if<!std::is_same<_Sseq, independent_bits_engine>::value && !std::is_same<_Sseq,
            _RandomNumberEngine>::value> ::type> std::independent_bits_engine< _RandomNumberEngine, __w,
            _UIntType >::independent_bits_engine ( _Sseq & __q ) [inline], [explicit]
```

Generator construct a independent_bits_engine engine.

Parameters

<code>__q</code>	A seed sequence.
------------------	------------------

Definition at line 1125 of file random.h.

5.781.4 Member Function Documentation

```
5.781.4.1 template<typename _RandomNumberEngine, size_t __w, typename _UIntType> const _RandomNumberEngine&
std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType>::base ( ) const [inline],
[noexcept]
```

Gets a const reference to the underlying generator engine object.

Definition at line 1160 of file random.h.

Referenced by std::operator<<().

```
5.781.4.2 template<typename _RandomNumberEngine, size_t __w, typename _UIntType> void std::independent_
_bits_engine<_RandomNumberEngine, __w, _UIntType>::discard ( unsigned long long __z )
[inline]
```

Discard a sequence of random numbers.

Definition at line 1181 of file random.h.

```
5.781.4.3 template<typename _RandomNumberEngine, size_t __w, typename _UIntType> static constexpr result_type
std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType>::max ( ) [inline],
[static]
```

Gets the maximum value in the generated random number range.

Definition at line 1174 of file random.h.

```
5.781.4.4 template<typename _RandomNumberEngine, size_t __w, typename _UIntType> static constexpr result_type
std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType>::min ( ) [inline], [static]
```

Gets the minimum value in the generated random number range.

Definition at line 1167 of file random.h.

```
5.781.4.5 template<typename _RandomNumberEngine, size_t __w, typename _UIntType> independent_bits_engine<
_RandomNumberEngine, __w, _UIntType>::result_type std::independent_bits_engine<
_RandomNumberEngine, __w, _UIntType>::operator() ( )
```

Gets the next value in the generated random number sequence.

Definition at line 745 of file bits/random.tcc.

References std::__lg(), std::numeric_limits<_Tp>::max(), std::numeric_limits<_Tp>::min(), and std::shuffle_order_↵
_engine<_RandomNumberEngine, __k>::operator()().

Referenced by std::discard_block_engine<_RandomNumberEngine, __p, __r>::operator()().

```
5.781.4.6 template<typename _RandomNumberEngine, size_t __w, typename _UIntType> void
std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType>::seed ( ) [inline]
```

Reseeds the independent_bits_engine object with the default seed for the underlying base class generator engine.

Definition at line 1134 of file random.h.

```
5.781.4.7 template<typename _RandomNumberEngine, size_t __w, typename _UIntType> void std::
::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >::seed ( result_type __s )
[inline]
```

Reseeds the independent_bits_engine object with the default seed for the underlying base class generator engine.

Definition at line 1142 of file random.h.

```
5.781.4.8 template<typename _RandomNumberEngine, size_t __w, typename _UIntType> template<typename _Sseq > void
std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >::seed ( _Sseq & __q ) [inline]
```

Reseeds the independent_bits_engine object with the given seed sequence.

Parameters

<code>__q</code>	A seed generator function.
------------------	----------------------------

Definition at line 1152 of file random.h.

5.781.5 Friends And Related Function Documentation

```
5.781.5.1 template<typename _RandomNumberEngine, size_t __w, typename _UIntType> bool operator==
( const independent_bits_engine< _RandomNumberEngine, __w, _UIntType > & __lhs, const
independent_bits_engine< _RandomNumberEngine, __w, _UIntType > & __rhs ) [friend]
```

Compares two independent_bits_engine random number generator objects of the same type for equality.

Parameters

<code>__lhs</code>	A independent_bits_engine random number generator object.
<code>__rhs</code>	Another independent_bits_engine random number generator object.

Returns

true if the infinite sequences of generated values would be equal, false otherwise.

Definition at line 1206 of file random.h.

```
5.781.5.2 template<typename _RandomNumberEngine, size_t __w, typename _UIntType> template<typename _CharT, typename
_Traits > std::basic_istream< _CharT, _Traits>& operator>> ( std::basic_istream< _CharT, _Traits > & __is,
std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > & __x ) [friend]
```

Extracts the current state of a % subtract_with_carry_engine random number generator engine __x from the input stream __is.

Parameters

<code>_is</code>	An input stream.
<code>_x</code>	A independent_bits_engine random number generator engine.

Returns

The input stream with the state of `_x` extracted or in an error state.

Definition at line 1224 of file random.h.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.782 std::indirect_array<_Tp> Class Template Reference

Public Types

- typedef `_Tp` **value_type**

Public Member Functions

- [indirect_array](#) (const [indirect_array](#) &)
- void [operator%=\(const valarray<_Tp> &\)](#) const
- template<class `_Dom` >
void [operator%=\(const _Expr<_Dom, _Tp> &\)](#) const
- void [operator&=\(const valarray<_Tp> &\)](#) const
- template<class `_Dom` >
void [operator&=\(const _Expr<_Dom, _Tp> &\)](#) const
- void [operator*=\(const valarray<_Tp> &\)](#) const
- template<class `_Dom` >
void [operator*=\(const _Expr<_Dom, _Tp> &\)](#) const
- void [operator+=\(const valarray<_Tp> &\)](#) const
- template<class `_Dom` >
void [operator+=\(const _Expr<_Dom, _Tp> &\)](#) const
- void [operator-=\(const valarray<_Tp> &\)](#) const
- template<class `_Dom` >
void [operator-=\(const _Expr<_Dom, _Tp> &\)](#) const
- void [operator/=\(const valarray<_Tp> &\)](#) const
- template<class `_Dom` >
void [operator/=\(const _Expr<_Dom, _Tp> &\)](#) const
- void [operator<=<const valarray<_Tp> &\)](#) const
- template<class `_Dom` >
void [operator<=<const _Expr<_Dom, _Tp> &\)](#) const

- [indirect_array](#) & [operator=](#) (const [indirect_array](#) &)
- void [operator=](#) (const [valarray](#)<_Tp> &) const
- void [operator=](#) (const _Tp &) const
- template<class _Dom >
void [operator=](#) (const _Expr<_Dom, _Tp> &) const
- void [operator>>=](#) (const [valarray](#)<_Tp> &) const
- template<class _Dom >
void [operator>>=](#) (const _Expr<_Dom, _Tp> &) const
- void [operator^=](#) (const [valarray](#)<_Tp> &) const
- template<class _Dom >
void [operator^=](#) (const _Expr<_Dom, _Tp> &) const
- void [operator|=](#) (const [valarray](#)<_Tp> &) const
- template<class _Dom >
void [operator|=](#) (const _Expr<_Dom, _Tp> &) const

Friends

- class [gslice_array](#)<_Tp>
- class [valarray](#)<_Tp>

5.782.1 Detailed Description

```
template<class Tp>
class std::indirect_array<_Tp>
```

Reference to arbitrary subset of an array.

An [indirect_array](#) is a reference to the actual elements of an array specified by an ordered array of indices. The way to get an [indirect_array](#) is to call [operator\[\]](#)([valarray](#)<size_t>) on a [valarray](#). The returned [indirect_array](#) then permits carrying operations out on the referenced subset of elements in the original [valarray](#).

For example, if an [indirect_array](#) is obtained using the array (4,2,0) as an argument, and then assigned to an array containing (1,2,3), then the underlying array will have `array[0]==3`, `array[2]==2`, and `array[4]==1`.

Parameters

<i>Tp</i>	Element type.
-----------	---------------

Definition at line 84 of file [valarray](#).

The documentation for this class was generated from the following files:

- [valarray](#)
- [indirect_array.h](#)

5.783 std::initializer_list<_E> Class Template Reference

Public Types

- typedef const _E * **const_iterator**
- typedef const _E & **const_reference**
- typedef const _E * **iterator**
- typedef const _E & **reference**
- typedef size_t **size_type**
- typedef _E **value_type**

Public Member Functions

- constexpr const_iterator **begin** () const noexcept
- constexpr const_iterator **end** () const noexcept
- constexpr size_type **size** () const noexcept

5.783.1 Detailed Description

```
template<class _E>  
class std::initializer_list<_E>
```

initializer_list

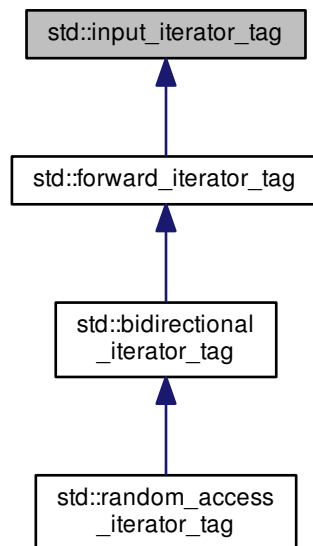
Definition at line 47 of file initializer_list.

The documentation for this class was generated from the following file:

- [initializer_list](#)

5.784 std::input_iterator_tag Struct Reference

Inheritance diagram for std::input_iterator_tag:



5.784.1 Detailed Description

Marking input iterators.

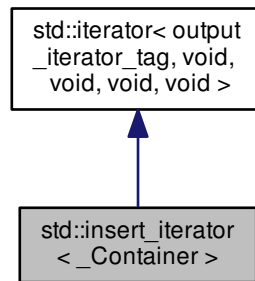
Definition at line 89 of file `stl_iterator_base_types.h`.

The documentation for this struct was generated from the following file:

- [stl_iterator_base_types.h](#)

5.785 `std::insert_iterator<_Container>` Class Template Reference

Inheritance diagram for `std::insert_iterator<_Container>`:



Public Types

- typedef `_Container` `container_type`
- typedef void `difference_type`
- typedef `output_iterator_tag` `iterator_category`
- typedef void `pointer`
- typedef void `reference`
- typedef void `value_type`

Public Member Functions

- `insert_iterator` (`_Container &__x`, `typename _Container::iterator __i`)
- `insert_iterator` & `operator*` ()
- `insert_iterator` & `operator++` ()
- `insert_iterator` & `operator++` (int)
- `insert_iterator` & `operator=` (const `typename _Container::value_type &__value`)
- `insert_iterator` & `operator=` (`typename _Container::value_type &&__value`)

Protected Attributes

- `_Container *` **`container`**
- `_Container::iterator` **`iter`**

5.785.1 Detailed Description

```
template<typename _Container>
class std::insert_iterator< _Container >
```

Turns assignment into insertion.

These are output iterators, constructed from a container-of-T. Assigning a T to the iterator inserts it in the container at the iterator's position, rather than overwriting the value at that position.

(Sequences will actually insert a *copy* of the value before the iterator's position.)

Tip: Using the inserter function to create these iterators can save typing.

Definition at line 636 of file bits/stl_iterator.h.

5.785.2 Member Typedef Documentation

5.785.2.1 `template<typename _Container> typedef _Container std::insert_iterator< _Container >::container_type`

A nested typedef for the type of whatever container you used.

Definition at line 645 of file bits/stl_iterator.h.

5.785.2.2 `typedef void std::iterator< output_iterator_tag , void , void , void , void >::difference_type` `[inherited]`

Distance between iterators is represented as this type.

Definition at line 125 of file stl_iterator_base_types.h.

5.785.2.3 `typedef output_iterator_tag std::iterator< output_iterator_tag , void , void , void , void >::iterator_category` `[inherited]`

One of the [tag types](#).

Definition at line 121 of file stl_iterator_base_types.h.

5.785.2.4 `typedef void std::iterator< output_iterator_tag , void , void , void , void >::pointer` `[inherited]`

This type represents a pointer-to-value_type.

Definition at line 127 of file stl_iterator_base_types.h.

5.785.2.5 `typedef void std::iterator< output_iterator_tag , void , void , void , void >::reference` `[inherited]`

This type represents a reference-to-value_type.

Definition at line 129 of file stl_iterator_base_types.h.

5.785.2.6 `typedef void std::iterator< output_iterator_tag , void , void , void , void >::value_type` `[inherited]`

The type "pointed to" by the iterator.

Definition at line 123 of file `stl_iterator_base_types.h`.

5.785.3 Constructor & Destructor Documentation

5.785.3.1 `template<typename _Container> std::insert_iterator< _Container >::insert_iterator (_Container & __x, typename _Container::iterator __i)` `[inline]`

The only way to create this iterator is with a container and an initial position (a normal iterator into the container).

Definition at line 651 of file `bits/stl_iterator.h`.

5.785.4 Member Function Documentation

5.785.4.1 `template<typename _Container> insert_iterator& std::insert_iterator< _Container >::operator* ()` `[inline]`

Simply returns `*this`.

Definition at line 705 of file `bits/stl_iterator.h`.

5.785.4.2 `template<typename _Container> insert_iterator& std::insert_iterator< _Container >::operator++ ()` `[inline]`

Simply returns `*this`. (This iterator does not *move*.)

Definition at line 710 of file `bits/stl_iterator.h`.

5.785.4.3 `template<typename _Container> insert_iterator& std::insert_iterator< _Container >::operator++ (int)` `[inline]`

Simply returns `*this`. (This iterator does not *move*.)

Definition at line 715 of file `bits/stl_iterator.h`.

5.785.4.4 `template<typename _Container> insert_iterator& std::insert_iterator< _Container >::operator= (const typename _Container::value_type & __value)` `[inline]`

Parameters

<code>__value</code>	An instance of whatever type <code>container_type::const_reference</code> is; presumably a reference-to-const T for <code>container<T></code> .
----------------------	---

Returns

This iterator, for chained operations.

This kind of iterator maintains its own position in the container. Assigning a value to the iterator will insert the value into the container at the place before the iterator.

The position is maintained such that subsequent assignments will insert values immediately after one another. For example,

```
// vector v contains A and Z
insert_iterator i (v, ++v.begin());
i = 1;
i = 2;
i = 3;

// vector v contains A, 1, 2, 3, and Z
```

Definition at line 687 of file bits/stl_iterator.h.

The documentation for this class was generated from the following file:

- [bits/stl_iterator.h](#)

5.786 std::integer_sequence<_Tp, _Idx > Struct Template Reference**Public Types**

- typedef **_Tp value_type**

Static Public Member Functions

- static constexpr **size_t size** ()

5.786.1 Detailed Description

```
template<typename _Tp, _Tp... _Idx>
struct std::integer_sequence<_Tp, _Idx >
```

Class template integer_sequence.

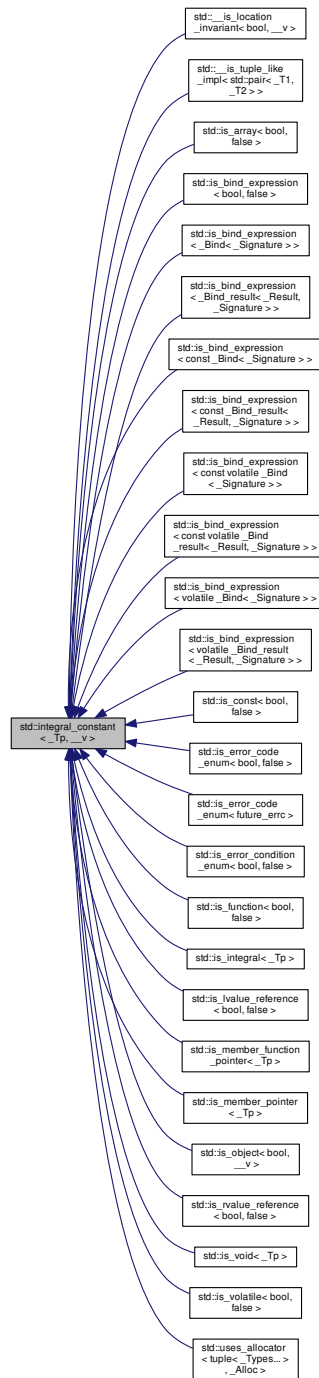
Definition at line 296 of file utility.

The documentation for this struct was generated from the following file:

- [utility](#)

5.787 std::integral_constant< _Tp, __v > Struct Template Reference

Inheritance diagram for std::integral_constant< _Tp, __v >:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr _Tp **value**

5.787.1 Detailed Description

```
template<typename _Tp, _Tp __v>  
struct std::integral_constant< _Tp, __v >
```

integral_constant

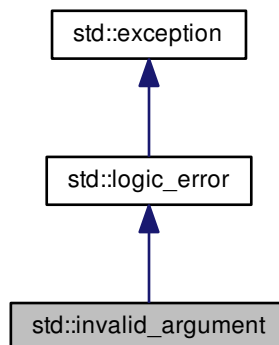
Definition at line 69 of file type_traits.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.788 std::invalid_argument Class Reference

Inheritance diagram for std::invalid_argument:



Public Member Functions

- **invalid_argument** (const [string](#) &__arg) _GLIBCXX_TXN_SAFE
- **invalid_argument** (const char *) _GLIBCXX_TXN_SAFE
- virtual const char * [what](#) () const _GLIBCXX_TXN_SAFE_DYN noexcept

5.788.1 Detailed Description

Thrown to report invalid arguments to functions.

Definition at line 158 of file stdexcept.

5.788.2 Member Function Documentation

5.788.2.1 virtual const char* std::logic_error::what () const [virtual], [noexcept], [inherited]

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

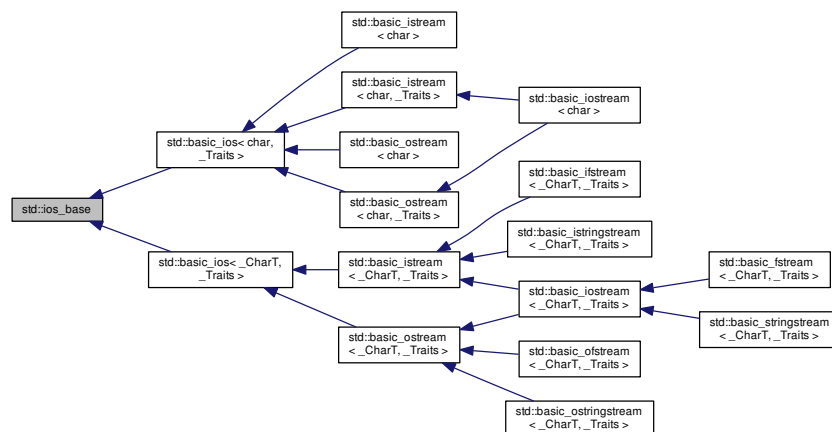
Reimplemented in [std::future_error](#).

The documentation for this class was generated from the following file:

- [stdexcept](#)

5.789 std::ios_base Class Reference

Inheritance diagram for std::ios_base:



Classes

- class [failure](#)

Public Types

- enum [event](#) { [erase_event](#), [imbue_event](#), [copyfmt_event](#) }
- typedef void(* [event_callback](#)) ([event](#) __e, [ios_base](#) &__b, int __i)
- typedef _ios_Fmtflags [fmtflags](#)
- typedef int [io_state](#)
- typedef _ios_istate [iostate](#)
- typedef int [open_mode](#)
- typedef _ios_Openmode [openmode](#)
- typedef int [seek_dir](#)
- typedef _ios_Seekdir [seekdir](#)
- typedef [std::streamoff](#) [streamoff](#)
- typedef [std::streampos](#) [streampos](#)

Public Member Functions

- [ios_base](#) (const [ios_base](#) &)=delete
- virtual [~ios_base](#) ()
- const [locale](#) & [_M_getloc](#) () const
- [fmtflags](#) [flags](#) () const
- [fmtflags](#) [flags](#) ([fmtflags](#) __fmtfl)
- [locale](#) [getloc](#) () const
- [locale](#) [imbue](#) (const [locale](#) &__loc) throw ()
- long & [iword](#) (int __ix)
- [ios_base](#) & [operator=](#) (const [ios_base](#) &)=delete
- [streamsize](#) [precision](#) () const
- [streamsize](#) [precision](#) ([streamsize](#) __prec)
- void *& [pword](#) (int __ix)
- void [register_callback](#) ([event_callback](#) __fn, int __index)
- [fmtflags](#) [setf](#) ([fmtflags](#) __fmtfl)
- [fmtflags](#) [setf](#) ([fmtflags](#) __fmtfl, [fmtflags](#) __mask)
- void [unsetf](#) ([fmtflags](#) __mask)
- [streamsize](#) [width](#) () const
- [streamsize](#) [width](#) ([streamsize](#) __wide)

Static Public Member Functions

- static bool [sync_with_stdio](#) (bool __sync=true)
- static int [xalloc](#) () throw ()

Static Public Attributes

- static const [fmtflags](#) `adjustfield`
- static const [openmode](#) `app`
- static const [openmode](#) `ate`
- static const [iostate](#) `badbit`
- static const [fmtflags](#) `basefield`
- static const [seekdir](#) `beg`
- static const [openmode](#) `binary`
- static const [fmtflags](#) `boolalpha`
- static const [seekdir](#) `cur`
- static const [fmtflags](#) `dec`
- static const [seekdir](#) `end`
- static const [iostate](#) `eofbit`
- static const [iostate](#) `failbit`
- static const [fmtflags](#) `fixed`
- static const [fmtflags](#) `floatfield`
- static const [iostate](#) `goodbit`
- static const [fmtflags](#) `hex`
- static const [openmode](#) `in`
- static const [fmtflags](#) `internal`
- static const [fmtflags](#) `left`
- static const [fmtflags](#) `oct`
- static const [openmode](#) `out`
- static const [fmtflags](#) `right`
- static const [fmtflags](#) `scientific`
- static const [fmtflags](#) `showbase`
- static const [fmtflags](#) `showpoint`
- static const [fmtflags](#) `showpos`
- static const [fmtflags](#) `skipws`
- static const [openmode](#) `trunc`
- static const [fmtflags](#) `unitbuf`
- static const [fmtflags](#) `uppercase`

Protected Types

- enum { [_S_local_word_size](#) }

Protected Member Functions

- void [_M_call_callbacks](#) ([event](#) __ev) throw ()
- void [_M_dispose_callbacks](#) (void) throw ()
- [_Words](#) & [_M_grow_words](#) (int __index, bool __iword)
- void [_M_init](#) () throw ()
- void [_M_move](#) ([ios_base](#) &) noexcept
- void [_M_swap](#) ([ios_base](#) &__rhs) noexcept

Protected Attributes

- `_Callback_list * _M_callbacks`
- `iosstate _M_exception`
- `fmtflags _M_flags`
- `locale _M_ios_locale`
- `_Words _M_local_word [_S_local_word_size]`
- `streamsize _M_precision`
- `iosstate _M_streambuf_state`
- `streamsize _M_width`
- `_Words * _M_word`
- `int _M_word_size`
- `_Words _M_word_zero`

5.789.1 Detailed Description

The base of the I/O class hierarchy.

This class defines everything that can be defined about I/O that does not depend on the type of characters being input or output. Most people will only see `ios_base` when they need to specify the full name of the various I/O flags (e.g., the openmodes).

Definition at line 228 of file `ios_base.h`.

5.789.2 Member Typedef Documentation

5.789.2.1 `typedef void(* std::ios_base::event_callback)(event __e, ios_base &__b, int __i)`

The type of an event callback function.

Parameters

<code>__e</code>	One of the members of the event enum.
<code>__b</code>	Reference to the <code>ios_base</code> object.
<code>__i</code>	The integer provided when the callback was registered.

Event callbacks are user defined functions that get called during several `ios_base` and `basic_ios` functions, specifically `imbue()`, `copyfmt()`, and `~ios()`.

Definition at line 504 of file `ios_base.h`.

5.789.2.2 `typedef _Ios_Fmtflags std::ios_base::fmtflags`

This is a bitmask type.

`_Ios_Fmtflags` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `fmtflags` are:

- `boolalpha`
- `dec`
- `fixed`
- `hex`
- `internal`
- `left`
- `oct`
- `right`
- `scientific`
- `showbase`
- `showpoint`
- `showpos`
- `skipws`
- `unitbuf`
- `uppercase`
- `adjustfield`
- `basefield`
- `floatfield`

Definition at line 323 of file `ios_base.h`.

5.789.2.3 `typedef _Ios_Iostate std::ios_base::iostate`

This is a bitmask type.

`_Ios_Iostate` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `iostate` are:

- `badbit`
- `eofbit`
- `failbit`
- `goodbit`

Definition at line 398 of file `ios_base.h`.

5.789.2.4 `typedef _Ios_Openmode std::ios_base::openmode`

This is a bitmask type.

`_Ios_Openmode` is implementation-defined, but it is valid to perform bitwise operations on these values and expect the Right Thing to happen. Defined objects of type `openmode` are:

- `app`
- `ate`
- `binary`
- `in`
- `out`
- `trunc`

Definition at line 429 of file `ios_base.h`.

5.789.2.5 `typedef _Ios_Seekdir std::ios_base::seekdir`

This is an enumerated type.

`_Ios_Seekdir` is implementation-defined. Defined values of type `seekdir` are:

- `beg`
- `cur`, equivalent to `SEEK_CUR` in the C standard library.
- `end`, equivalent to `SEEK_END` in the C standard library.

Definition at line 461 of file `ios_base.h`.

5.789.3 Member Enumeration Documentation

5.789.3.1 `enum std::ios_base::event`

The set of events that may be passed to an event callback.

`erase_event` is used during `~ios()` and `copyfmt()`. `imbue_event` is used during `imbue()`. `copyfmt_event` is used during `copyfmt()`.

Definition at line 487 of file `ios_base.h`.

5.789.4 Constructor & Destructor Documentation

5.789.4.1 virtual std::ios_base::~ios_base () [virtual]

Invokes each callback with erase_event. Destroys local storage.

Note that the ios_base object for the standard streams never gets destroyed. As a result, any callbacks registered with the standard streams will not get invoked with erase_event (unless copyfmt is used).

5.789.5 Member Function Documentation

5.789.5.1 const locale& std::ios_base::_M_getloc () const [inline]

Locale access.

Returns

A reference to the current locale.

Like getloc above, but returns a reference instead of generating a copy.

Definition at line 774 of file ios_base.h.

Referenced by std::time_get< _CharT, _Inlter >::do_date_order(), std::time_get< _CharT, _Inlter >::do_get(), std::money_get< _CharT, _Inlter >::do_get(), std::num_get< _CharT, _Inlter >::do_get(), std::time_get< _CharT, _Inlter >::do_get_date(), std::time_get< _CharT, _Inlter >::do_get_monthname(), std::time_get< _CharT, _Inlter >::do_get_time(), std::time_get< _CharT, _Inlter >::do_get_weekday(), std::time_get< _CharT, _Inlter >::do_get_year(), std::time_put< _CharT, _Outlter >::do_put(), std::num_put< _CharT, _Outlter >::do_put(), std::time_get< _CharT, _Inlter >::get(), and std::time_put< _CharT, _Outlter >::put().

5.789.5.2 fmtflags std::ios_base::flags () const [inline]

Access to format flags.

Returns

The format control flags for both input and output.

Definition at line 619 of file ios_base.h.

Referenced by std::basic_ios< _CharT, _Traits >::copyfmt(), std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >::discard(), std::money_get< _CharT, _Inlter >::do_get(), std::num_get< _CharT, _Inlter >::do_get(), std::num_put< _CharT, _Outlter >::do_put(), std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::operator>(), std::discard_block_engine< _RandomNumberEngine, __p, __r >::operator(), std::shuffle_order_engine< _RandomNumberEngine, __k >::operator(), std::normal_distribution< _RealType >::operator(), std::gamma_distribution< _RealType >::operator(), std::binomial_distribution< _IntType >::operator(), std::negative_binomial_distribution< _IntType >::operator(), std::poisson_distribution< _IntType >::operator(), std::basic_ostream< _CharT, _Traits >::operator<<(), std::operator<<(), std::__detail::operator>>(), std::operator>>(), std::linear_congruential_engine< _UIntType, __a, __c, __m >::seed(), and std::basic_istream< _CharT, _Traits >::sentry::sentry().

5.789.5.3 fmtflags std::ios_base::flags (fmtflags __fmtfl) [inline]

Setting new format flags all at once.

Parameters

<code>__fmtfl</code>	The new flags to set.
----------------------	-----------------------

Returns

The previous format control flags.

This function overwrites all the format flags with `__fmtfl`.

Definition at line 630 of file `ios_base.h`.

5.789.5.4 locale `std::ios_base::getloc () const [inline]`

Locale access.

Returns

A copy of the current locale.

If `imbue(loc)` has previously been called, then this function returns `loc`. Otherwise, it returns a copy of `std::locale()`, the global C++ locale.

Definition at line 763 of file `ios_base.h`.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::money_put<_CharT, _OutIter>::do_put()`, `std::operator>>()`, and `std::ws()`.

5.789.5.5 locale `std::ios_base::imbue (const locale & __loc) throw ()`

Setting a new locale.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Sets the new locale for this stream, and then invokes each callback with `imbue_event`.

Referenced by `std::basic_ios<_CharT, _Traits>::imbue()`.

5.789.5.6 `long& std::ios_base::iword (int __ix) [inline]`

Access to integer array.

Parameters

<code>_↔ _ix</code>	Index into the array.
-------------------------	-----------------------

Returns

A reference to an integer associated with the index.

The `iword` function provides access to an array of integers that can be used for any purpose. The array grows as required to hold the supplied index. All integers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 809 of file `ios_base.h`.

5.789.5.7 `streamsize std::ios_base::precision () const` `[inline]`

Flags access.

Returns

The precision to generate on certain output operations.

Be careful if you try to give a definition of *precision* here; see DR 189.

Definition at line 689 of file `ios_base.h`.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::normal_↔distribution<_RealType>::operator()()`, `std::gamma_distribution<_RealType>::operator()()`, `std::negative_binomial_↔distribution<_IntType>::operator()()`, and `std::operator>>()`.

5.789.5.8 `streamsize std::ios_base::precision (streamsize __prec)` `[inline]`

Changing flags.

Parameters

<code>__prec</code>	The new precision value.
---------------------	--------------------------

Returns

The previous value of `precision()`.

Definition at line 698 of file `ios_base.h`.

5.789.5.9 `void*& std::ios_base::pword (int __ix) [inline]`

Access to void pointer array.

Parameters

<code>__ix</code>	Index into the array.
-------------------	-----------------------

Returns

A reference to a `void*` associated with the index.

The `pword` function provides access to an array of pointers that can be used for any purpose. The array grows as required to hold the supplied index. All pointers in the array are initialized to 0.

The implementation reserves several indices. You should use `xalloc` to obtain an index that is safe to use. Also note that since the array can grow dynamically, it is not safe to hold onto the reference.

Definition at line 830 of file `ios_base.h`.

5.789.5.10 `void std::ios_base::register_callback (event_callback __fn, int __index)`

Add the callback `__fn` with parameter `__index`.

Parameters

<code>__fn</code>	The function to add.
<code>__index</code>	The integer to pass to the function when invoked.

Registers a function as an event callback with an integer parameter to be passed to the function when invoked. Multiple copies of the function are allowed. If there are multiple callbacks, they are invoked in the order they were registered.

5.789.5.11 `fmtflags std::ios_base::setf (fmtflags __fmtfl) [inline]`

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
----------------------	--------------------------

Returns

The previous format control flags.

This function sets additional flags in format control. Flags that were previously set remain set.

Definition at line 646 of file ios_base.h.

Referenced by `std::boolalpha()`, `std::dec()`, `std::fixed()`, `std::hex()`, `std::hexfloat()`, `std::left()`, `std::oct()`, `std::__detail::operator>>()`, `std::right()`, `std::scientific()`, `std::showbase()`, `std::showpoint()`, `std::showpos()`, `std::skipws()`, `std::unitbuf()`, and `std::uppercase()`.

5.789.5.12 `fmtflags std::ios_base::setf (fmtflags __fmtfl, fmtflags __mask)` `[inline]`

Setting new format flags.

Parameters

<code>__fmtfl</code>	Additional flags to set.
<code>__mask</code>	The flags mask for <code>fmtfl</code> .

Returns

The previous format control flags.

This function clears `mask` in the format flags, then sets `fmtfl` & `mask`. An example mask is `ios_base::adjustfield`.

Definition at line 663 of file ios_base.h.

5.789.5.13 `static bool std::ios_base::sync_with_stdio (bool __sync = true)` `[static]`

Interaction with the standard C I/O objects.

Parameters

<code>__sync</code>	Whether to synchronize or not.
---------------------	--------------------------------

Returns

True if the standard streams were previously synchronized.

The synchronization referred to is *only* that between the standard C facilities (e.g., `stdout`) and the standard C++ objects (e.g., `cout`). User-declared streams are unaffected. See <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>

5.789.5.14 `void std::ios_base::unsetf (fmtflags __mask)` `[inline]`

Clearing format flags.

Parameters

<code>__mask</code>	The flags to unset.
---------------------	---------------------

This function clears `__mask` in the format flags.

Definition at line 678 of file `ios_base.h`.

Referenced by `std::defaultfloat()`, `std::noboolalpha()`, `std::noshowbase()`, `std::noshowpoint()`, `std::noshowpos()`, `std::noskipws()`, `std::nounitbuf()`, and `std::nouppercase()`.

5.789.5.15 `streamsize std::ios_base::width () const` `[inline]`

Flags access.

Returns

The minimum field width to generate on output operations.

Minimum field width refers to the number of characters.

Definition at line 712 of file `ios_base.h`.

Referenced by `std::basic_ios<_CharT, _Traits>::copyfmt()`, `std::money_get<_CharT, _InIter>::do_get()`, `std::num_get<_CharT, _InIter>::do_get()`, `std::num_put<_CharT, _OutIter>::do_put()`, and `std::operator>>()`.

5.789.5.16 `streamsize std::ios_base::width (streamsize __wide)` `[inline]`

Changing flags.

Parameters

<code>__wide</code>	The new width value.
---------------------	----------------------

Returns

The previous value of `width()`.

Definition at line 721 of file `ios_base.h`.

5.789.5.17 `static int std::ios_base::xalloc () throw` `[static]`

Access to unique indices.

Returns

An integer different from all previous calls.

This function returns a unique integer every time it is called. It can be used for any purpose, but is primarily intended to be a unique index for the `iwor`d and `pwor`d functions. The expectation is that an application calls `xalloc` in order to obtain an index in the `iwor`d and `pwor`d arrays that can be used without fear of conflict.

The implementation maintains a static variable that is incremented and returned on each invocation. `xalloc` is guaranteed to return an index that is safe to use in the `iwor`d and `pwor`d arrays.

5.789.6 Member Data Documentation

5.789.6.1 const fmtflags std::ios_base::adjustfield [static]

A mask of left|right|internal. Useful for the 2-arg form of `setf`.

Definition at line 378 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::internal()`, `std::left()`, and `std::right()`.

5.789.6.2 const openmode std::ios_base::app [static]

Seek to end before each write.

Definition at line 432 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::overflow()`, and `std::basic_filebuf< _CharT, _Traits >::xsputn()`.

5.789.6.3 const openmode std::ios_base::ate [static]

Open and seek to end immediately after opening.

Definition at line 435 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::open()`.

5.789.6.4 const iostate std::ios_base::badbit [static]

Indicates a loss of integrity in an input or output sequence (such as an irrecoverable read error from a file).

Definition at line 402 of file `ios_base.h`.

Referenced by `std::basic_ios< char, _Traits >::bad()`, `std::basic_ios< char, _Traits >::fail()`, `std::basic_ostream< _CharT, _Traits >::flush()`, `std::basic_istream< _CharT, _Traits >::get()`, `std::basic_istream< _CharT, _Traits >::getline()`, `std::basic_istream< _CharT, _Traits >::ignore()`, `std::basic_ostream< _CharT, _Traits >::operator<<()`, `std::operator<<()`, `std::basic_istream< _CharT, _Traits >::operator>>()`, `std::operator>>()`, `std::basic_istream< _CharT, _Traits >::peek()`, `std::basic_ostream< _CharT, _Traits >::put()`, `std::basic_istream< _CharT, _Traits >::putback()`, `std::basic_istream< _CharT, _Traits >::read()`, `std::basic_istream< _CharT, _Traits >::readsome()`, `std::basic_istream< _CharT, _Traits >::seekg()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_ostream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, `std::basic_istream< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::tellg()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, `std::basic_istream< _CharT, _Traits >::unget()`, and `std::basic_ostream< _CharT, _Traits >::write()`.

5.789.6.5 const fmtflags std::ios_base::basefield [static]

A mask of dec|oct|hex. Useful for the 2-arg form of `setf`.

Definition at line 381 of file `ios_base.h`.

Referenced by `std::dec()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::hex()`, `std::oct()`, and `std::basic_ostream< _CharT, _Traits >::operator<<()`.

5.789.6.6 `const seekdir std::ios_base::beg` `[static]`

Request a seek relative to the beginning of the stream.

Definition at line 464 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::seekpos()`, and `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync()`.

5.789.6.7 `const openmode std::ios_base::binary` `[static]`

Perform input and output in binary mode (as opposed to text mode). This is probably not what you think it is; see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/fstreams.html#std.io.filestreams.binary>.

Definition at line 440 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::showmanyc()`.

5.789.6.8 `const fmtflags std::ios_base::boolalpha` `[static]`

Insert/extract `bool` in alphabetic rather than numeric format.

Definition at line 326 of file `ios_base.h`.

Referenced by `std::boolalpha()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, and `std::noboolalpha()`.

5.789.6.9 `const seekdir std::ios_base::cur` `[static]`

Request a seek relative to the current position within the sequence.

Definition at line 467 of file `ios_base.h`.

Referenced by `std::basic_filebuf< _CharT, _Traits >::imbue()`, `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_filebuf< _CharT, _Traits >::pbackfail()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`, `std::basic_filebuf< _CharT, _Traits >::seekoff()`, `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync()`, `std::basic_istream< _CharT, _Traits >::tellg()`, and `std::basic_ostream< _CharT, _Traits >::tellp()`.

5.789.6.10 `const fmtflags std::ios_base::dec` `[static]`

Converts integer input or generates integer output in decimal base.

Definition at line 329 of file `ios_base.h`.

Referenced by `std::dec()`.

5.789.6.11 const seekdir std::ios_base::end [static]

Request a seek relative to the current end of the sequence.

Definition at line 470 of file ios_base.h.

Referenced by std::basic_filebuf< _CharT, _Traits >::open(), and std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff().

5.789.6.12 const iostate std::ios_base::eofbit [static]

Indicates that an input operation reached the end of an input sequence.

Definition at line 405 of file ios_base.h.

Referenced by std::time_get< _CharT, _Inlter >::do_get(), std::num_get< _CharT, _Inlter >::do_get(), std::time_get< _CharT, _Inlter >::do_get_date(), std::time_get< _CharT, _Inlter >::do_get_monthname(), std::time_get< _CharT, _Inlter >::do_get_time(), std::time_get< _CharT, _Inlter >::do_get_weekday(), std::time_get< _CharT, _Inlter >::do_get_year(), std::basic_ios< char, _Traits >::eof(), std::basic_istream< _CharT, _Traits >::get(), std::time_get< _CharT, _Inlter >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::basic_istream< _CharT, _Traits >::ignore(), std::basic_istream< _CharT, _Traits >::operator>>(), std::operator>>(), std::basic_istream< _CharT, _Traits >::peek(), std::basic_istream< _CharT, _Traits >::putback(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::readsome(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_istream< _CharT, _Traits >::sentry::sentry(), std::basic_istream< _CharT, _Traits >::unget(), and std::ws().

5.789.6.13 const iostate std::ios_base::failbit [static]

Indicates that an input operation failed to read the expected characters, or that an output operation failed to generate the desired characters.

Definition at line 410 of file ios_base.h.

Referenced by std::time_get< _CharT, _Inlter >::do_date_order(), std::num_get< _CharT, _Inlter >::do_get(), std::time_get< _CharT, _Inlter >::do_get_monthname(), std::time_get< _CharT, _Inlter >::do_get_weekday(), std::time_get< _CharT, _Inlter >::do_get_year(), std::basic_ios< char, _Traits >::fail(), std::basic_istream< _CharT, _Traits >::get(), std::time_get< _CharT, _Inlter >::get(), std::basic_istream< _CharT, _Traits >::getline(), std::basic_ostream< _CharT, _Traits >::operator<<(), std::basic_istream< _CharT, _Traits >::operator>>(), std::operator>>(), std::basic_istream< _CharT, _Traits >::read(), std::basic_istream< _CharT, _Traits >::seekg(), std::basic_ostream< _CharT, _Traits >::seekp(), std::basic_ostream< _CharT, _Traits >::sentry::sentry(), and std::basic_istream< _CharT, _Traits >::sentry::sentry().

5.789.6.14 const fmtflags std::ios_base::fixed [static]

Generate floating-point output in fixed-point notation.

Definition at line 332 of file ios_base.h.

Referenced by std::num_get< _CharT, _Inlter >::do_get(), std::fixed(), and std::hexfloat().

5.789.6.15 `const fmtflags std::ios_base::floatfield` `[static]`

A mask of scientific|fixed. Useful for the 2-arg form of `setf`.

Definition at line 384 of file `ios_base.h`.

Referenced by `std::defaultfloat()`, `std::num_get<_CharT, _Inlter>::do_get()`, `std::fixed()`, `std::hexfloat()`, and `std::scientific()`.

5.789.6.16 `const iostate std::ios_base::goodbit` `[static]`

Indicates all is well.

Definition at line 413 of file `ios_base.h`.

Referenced by `std::time_get<_CharT, _Inlter>::do_date_order()`, `std::time_get<_CharT, _Inlter>::do_get()`, `std::num_get<_CharT, _Inlter>::do_get()`, `std::time_get<_CharT, _Inlter>::do_get_monthname()`, `std::time_get<_CharT, _Inlter>::do_get_weekday()`, `std::time_get<_CharT, _Inlter>::do_get_year()`, `std::basic_ostream<_CharT, _Traits>::flush()`, `std::basic_istream<_CharT, _Traits>::get()`, `std::time_get<_CharT, _Inlter>::get()`, `std::basic_istream<_CharT, _Traits>::getline()`, `std::basic_istream<_CharT, _Traits>::ignore()`, `std::basic_ostream<_CharT, _Traits>::operator<<()`, `std::basic_istream<_CharT, _Traits>::operator>>()`, `std::operator>>()`, `std::basic_istream<_CharT, _Traits>::peek()`, `std::basic_ostream<_CharT, _Traits>::put()`, `std::basic_istream<_CharT, _Traits>::putback()`, `std::basic_ios<char, _Traits>::rdstate()`, `std::basic_istream<_CharT, _Traits>::read()`, `std::basic_istream<_CharT, _Traits>::readsome()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_ostream<_CharT, _Traits>::seekp()`, `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`, `std::basic_istream<_CharT, _Traits>::sentry::sentry()`, `std::basic_istream<_CharT, _Traits>::sync()`, and `std::basic_istream<_CharT, _Traits>::unset()`.

5.789.6.17 `const fmtflags std::ios_base::hex` `[static]`

Converts integer input or generates integer output in hexadecimal base.

Definition at line 335 of file `ios_base.h`.

Referenced by `std::num_get<_CharT, _Inlter>::do_get()`, `std::num_put<_CharT, _Outlter>::do_put()`, `std::hex()`, and `std::basic_ostream<_CharT, _Traits>::operator<<()`.

5.789.6.18 `const openmode std::ios_base::in` `[static]`

Open for input. Default for `ifstream` and `fstream`.

Definition at line 443 of file `ios_base.h`.

Referenced by `std::basic_filebuf<_CharT, _Traits>::pbackfail()`, `std::basic_istream<_CharT, _Traits>::seekg()`, `std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekoff()`, `std::basic_stringbuf<_CharT, _Traits, _Alloc>::seekpos()`, `std::basic_filebuf<_CharT, _Traits>::showmanyc()`, `__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>::sync()`, `std::basic_istream<_CharT, _Traits>::tellg()`, `std::basic_stringbuf<_CharT, _Traits, _Alloc>::underflow()`, `std::basic_filebuf<_CharT, _Traits>::underflow()`, and `std::basic_filebuf<_CharT, _Traits>::xsgetn()`.

5.789.6.19 const fmtflags std::ios_base::internal [static]

Adds fill characters at a designated internal point in certain generated output, or identical to `right` if no such point is designated.

Definition at line 340 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, and `std::ios_base::internal()`.

5.789.6.20 const fmtflags std::ios_base::left [static]

Adds fill characters on the right (final positions) of certain generated output. (I.e., the thing you print is flush left.)

Definition at line 344 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, and `std::ios_base::left()`.

5.789.6.21 const fmtflags std::ios_base::oct [static]

Converts integer input or generates integer output in octal base.

Definition at line 347 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _InIter >::do_get()`, `std::oct()`, and `std::basic_ostream< _CharT, _Traits >::operator<<()`.

5.789.6.22 const openmode std::ios_base::out [static]

Open for output. Default for `ofstream` and `fstream`.

Definition at line 446 of file `ios_base.h`.

Referenced by `std::basic_stringbuf< _CharT, _Traits, _Alloc >::overflow()`, `std::basic_filebuf< _CharT, _Traits >::overflow()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::pbackfail()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekoff()`, `std::basic_ostream< _CharT, _Traits >::seekp()`, `std::basic_stringbuf< _CharT, _Traits, _Alloc >::seekpos()`, `__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >::sync()`, `std::basic_ostream< _CharT, _Traits >::tellp()`, and `std::basic_filebuf< _CharT, _Traits >::xspn()`.

5.789.6.23 const fmtflags std::ios_base::right [static]

Adds fill characters on the left (initial positions) of certain generated output. (I.e., the thing you print is flush right.)

Definition at line 351 of file `ios_base.h`.

Referenced by `std::right()`.

5.789.6.24 `const fmtflags std::ios_base::scientific` `[static]`

Generates floating-point output in scientific notation.

Definition at line 354 of file `ios_base.h`.

Referenced by `std::hexfloat()`, and `std::scientific()`.

5.789.6.25 `const fmtflags std::ios_base::showbase` `[static]`

Generates a prefix indicating the numeric base of generated integer output.

Definition at line 358 of file `ios_base.h`.

Referenced by `std::money_get< _CharT, _InIter >::do_get()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::noshowbase()`, and `std::showbase()`.

5.789.6.26 `const fmtflags std::ios_base::showpoint` `[static]`

Generates a decimal-point character unconditionally in generated floating-point output.

Definition at line 362 of file `ios_base.h`.

Referenced by `std::noshowpoint()`, and `std::showpoint()`.

5.789.6.27 `const fmtflags std::ios_base::showpos` `[static]`

Generates a + sign in non-negative generated numeric output.

Definition at line 365 of file `ios_base.h`.

Referenced by `std::num_get< _CharT, _InIter >::do_get()`, `std::noshowpos()`, and `std::showpos()`.

5.789.6.28 `const fmtflags std::ios_base::skipws` `[static]`

Skips leading white space before certain input operations.

Definition at line 368 of file `ios_base.h`.

Referenced by `std::noskipws()`, `std::basic_istream< _CharT, _Traits >::sentry::sentry()`, and `std::skipws()`.

5.789.6.29 `const openmode std::ios_base::trunc` `[static]`

Open for input. Default for `ofstream`.

Definition at line 449 of file `ios_base.h`.

5.790.1 Detailed Description

These are thrown to indicate problems with io.

27.4.2.1.1 Class `ios_base::failure`.

Definition at line 276 of file `ios_base.h`.

5.790.2 Member Function Documentation

5.790.2.1 `virtual const char* std::ios_base::failure::what () const throw ()` `[virtual]`

Returns a C-style character string describing the general cause of the current error.

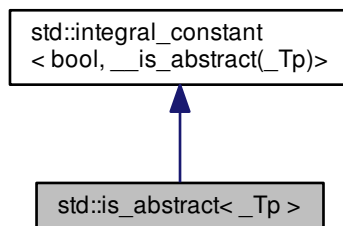
Reimplemented from [std::exception](#).

The documentation for this class was generated from the following file:

- [ios_base.h](#)

5.791 `std::is_abstract<_Tp>` Struct Template Reference

Inheritance diagram for `std::is_abstract<_Tp>`:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr bool **value**

5.791.1 Detailed Description

```
template<typename _Tp>  
struct std::is_abstract<_Tp>
```

`is_abstract`

Definition at line 722 of file `type_traits`.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.792 `std::is_arithmetic<_Tp>` Struct Template Reference

Inherits type< `is_integral<_Tp>`, `is_floating_point<_Tp>` >.

5.792.1 Detailed Description

```
template<typename _Tp>  
struct std::is_arithmetic<_Tp>
```

`is_arithmetic`

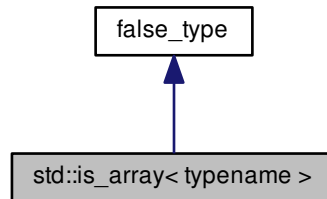
Definition at line 584 of file `type_traits`.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.793 `std::is_array< typename >` Struct Template Reference

Inheritance diagram for `std::is_array< typename >`:



Public Types

- typedef [integral_constant](#)< `_Tp`, `__v` > **type**
- typedef `_Tp` **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr `_Tp` **value**

5.793.1 Detailed Description

```
template<typename>
struct std::is_array< typename >
```

`is_array`

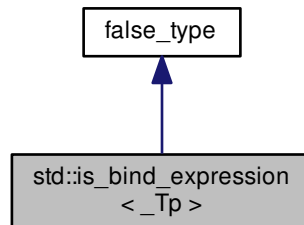
Definition at line 356 of file `type_traits`.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.794 `std::is_bind_expression< _Tp >` Struct Template Reference

Inheritance diagram for `std::is_bind_expression< _Tp >`:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr _Tp **value**

5.794.1 Detailed Description

```
template<typename _Tp>
struct std::is_bind_expression< _Tp >
```

Determines if the given type `_Tp` is a function object that should be treated as a subexpression when evaluating calls to function objects returned by `bind()`.

C++11 [func.bind.isbind].

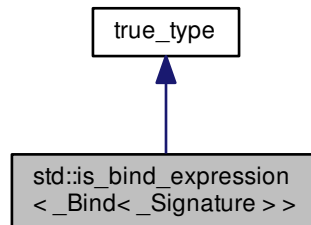
Definition at line 661 of file `functional`.

The documentation for this struct was generated from the following file:

- [functional](#)

5.795 `std::is_bind_expression<_Bind<_Signature>>` Struct Template Reference

Inheritance diagram for `std::is_bind_expression<_Bind<_Signature>>`:



Public Types

- typedef [integral_constant](#)<_Tp, __v> **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr _Tp **value**

5.795.1 Detailed Description

```
template<typename _Signature>
struct std::is_bind_expression<_Bind<_Signature>>>
```

Class template `_Bind` is always a bind expression.

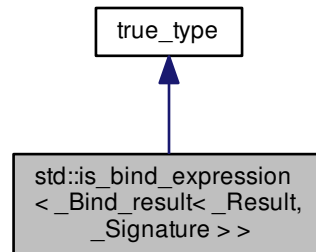
Definition at line 1205 of file `functional`.

The documentation for this struct was generated from the following file:

- [functional](#)

5.796 `std::is_bind_expression< _Bind_result< _Result, _Signature > >` Struct Template Reference

Inheritance diagram for `std::is_bind_expression< _Bind_result< _Result, _Signature > >`:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr _Tp **value**

5.796.1 Detailed Description

```

template<typename _Result, typename _Signature>
struct std::is_bind_expression< _Bind_result< _Result, _Signature > >
  
```

Class template `_Bind_result` is always a bind expression.

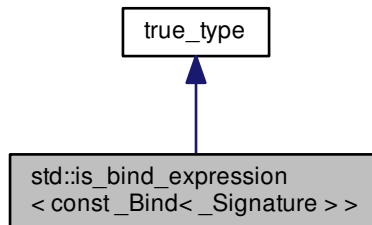
Definition at line 1237 of file `functional`.

The documentation for this struct was generated from the following file:

- [functional](#)

5.797 `std::is_bind_expression< const _Bind< _Signature > >` Struct Template Reference

Inheritance diagram for `std::is_bind_expression< const _Bind< _Signature > >`:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr _Tp **value**

5.797.1 Detailed Description

```
template<typename _Signature>
struct std::is_bind_expression< const _Bind< _Signature > >
```

Class template `_Bind` is always a bind expression.

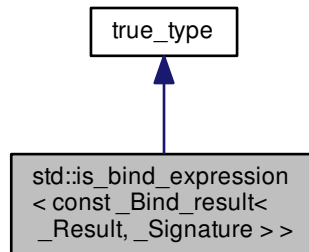
Definition at line 1213 of file `functional`.

The documentation for this struct was generated from the following file:

- [functional](#)

5.798 `std::is_bind_expression< const _Bind_result< _Result, _Signature > >` Struct Template Reference

Inheritance diagram for `std::is_bind_expression< const _Bind_result< _Result, _Signature > >`:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr _Tp **value**

5.798.1 Detailed Description

```

template<typename _Result, typename _Signature>
struct std::is_bind_expression< const _Bind_result< _Result, _Signature > >

```

Class template `_Bind_result` is always a bind expression.

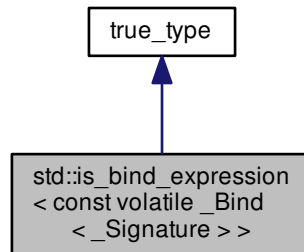
Definition at line 1245 of file `functional`.

The documentation for this struct was generated from the following file:

- [functional](#)

5.799 `std::is_bind_expression< const volatile _Bind< _Signature > >` Struct Template Reference

Inheritance diagram for `std::is_bind_expression< const volatile _Bind< _Signature > >`:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr _Tp **value**

5.799.1 Detailed Description

```
template<typename _Signature>
struct std::is_bind_expression< const volatile _Bind< _Signature > >
```

Class template `_Bind` is always a bind expression.

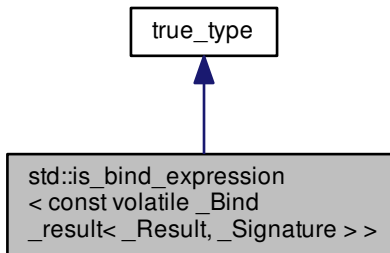
Definition at line 1229 of file `functional`.

The documentation for this struct was generated from the following file:

- [functional](#)

5.800 `std::is_bind_expression< const volatile _Bind_result< _Result, _Signature > >` Struct Template Reference

Inheritance diagram for `std::is_bind_expression< const volatile _Bind_result< _Result, _Signature > >`:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr _Tp **value**

5.800.1 Detailed Description

```
template<typename _Result, typename _Signature>  
struct std::is_bind_expression< const volatile _Bind_result< _Result, _Signature > >
```

Class template `_Bind_result` is always a bind expression.

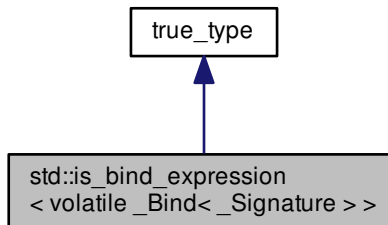
Definition at line 1261 of file `functional`.

The documentation for this struct was generated from the following file:

- [functional](#)

5.801 `std::is_bind_expression< volatile _Bind< _Signature > >` Struct Template Reference

Inheritance diagram for `std::is_bind_expression< volatile _Bind< _Signature > >`:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr _Tp **value**

5.801.1 Detailed Description

```
template<typename _Signature>
struct std::is_bind_expression< volatile _Bind< _Signature > >
```

Class template `_Bind` is always a bind expression.

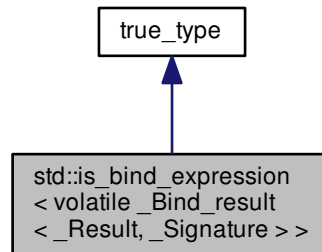
Definition at line 1221 of file `functional`.

The documentation for this struct was generated from the following file:

- [functional](#)

5.802 `std::is_bind_expression< volatile _Bind_result< _Result, _Signature > >` Struct Template Reference

Inheritance diagram for `std::is_bind_expression< volatile _Bind_result< _Result, _Signature > >`:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr _Tp **value**

5.802.1 Detailed Description

```

template<typename _Result, typename _Signature>
struct std::is_bind_expression< volatile _Bind_result< _Result, _Signature > >

```

Class template `_Bind_result` is always a bind expression.

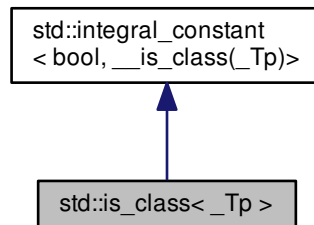
Definition at line 1253 of file `functional`.

The documentation for this struct was generated from the following file:

- [functional](#)

5.803 `std::is_class<_Tp>` Struct Template Reference

Inheritance diagram for `std::is_class<_Tp>`:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr bool **value**

5.803.1 Detailed Description

```
template<typename _Tp>
struct std::is_class<_Tp>
```

`is_class`

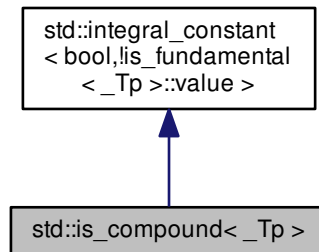
Definition at line 446 of file `type_traits`.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.804 std::is_compound<_Tp> Struct Template Reference

Inheritance diagram for std::is_compound<_Tp>:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr bool **value**

5.804.1 Detailed Description

```
template<typename _Tp>  
struct std::is_compound<_Tp>
```

is_compound

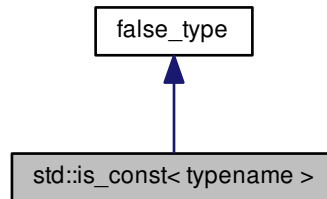
Definition at line 614 of file type_traits.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.805 `std::is_const< typename >` Struct Template Reference

Inheritance diagram for `std::is_const< typename >`:



Public Types

- typedef [integral_constant](#)< `_Tp`, `__v` > **type**
- typedef `_Tp` **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr `_Tp` **value**

5.805.1 Detailed Description

```
template<typename>
struct std::is_const< typename >
```

`is_const`

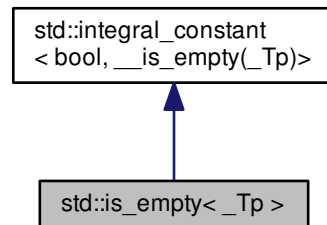
Definition at line 652 of file `type_traits`.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.806 std::is_empty< _Tp > Struct Template Reference

Inheritance diagram for std::is_empty< _Tp >:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr bool **value**

5.806.1 Detailed Description

```
template<typename _Tp>  
struct std::is_empty< _Tp >
```

is_empty

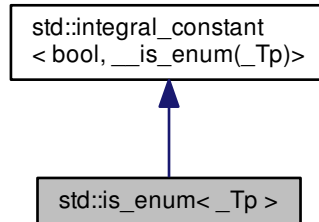
Definition at line 701 of file type_traits.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.807 std::is_enum< _Tp > Struct Template Reference

Inheritance diagram for std::is_enum< _Tp >:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr bool **value**

5.807.1 Detailed Description

```
template<typename _Tp>
struct std::is_enum< _Tp >
```

is_enum

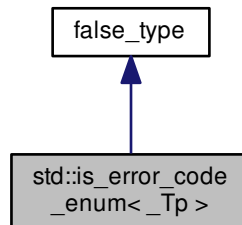
Definition at line 434 of file type_traits.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.808 std::is_error_code_enum< _Tp > Struct Template Reference

Inheritance diagram for std::is_error_code_enum< _Tp >:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr _Tp **value**

5.808.1 Detailed Description

```
template<typename _Tp>  
struct std::is_error_code_enum< _Tp >
```

is_error_code_enum

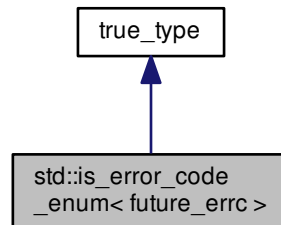
Definition at line 53 of file system_error.

The documentation for this struct was generated from the following file:

- [system_error](#)

5.809 `std::is_error_code_enum< future_errc >` Struct Template Reference

Inheritance diagram for `std::is_error_code_enum< future_errc >`:



Public Types

- typedef [integral_constant](#)< `_Tp`, `__v` > **type**
- typedef `_Tp` **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr `_Tp` **value**

5.809.1 Detailed Description

```
template<>
struct std::is_error_code_enum< future_errc >
```

Specialization.

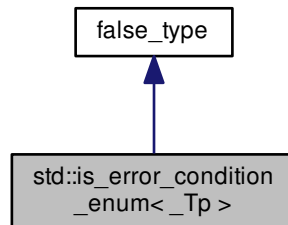
Definition at line 75 of file `future`.

The documentation for this struct was generated from the following file:

- [future](#)

5.810 `std::is_error_condition_enum< _Tp >` Struct Template Reference

Inheritance diagram for `std::is_error_condition_enum< _Tp >`:



Public Types

- typedef [integral_constant< _Tp, __v >](#) **type**
- typedef `_Tp` **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr `_Tp` **value**

5.810.1 Detailed Description

```
template<typename _Tp>  
struct std::is_error_condition_enum< _Tp >
```

`is_error_condition_enum`

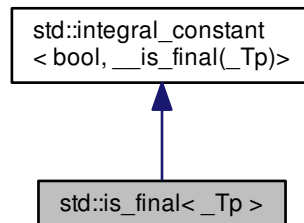
Definition at line 57 of file `system_error`.

The documentation for this struct was generated from the following file:

- [system_error](#)

5.811 `std::is_final<_Tp>` Struct Template Reference

Inheritance diagram for `std::is_final<_Tp>`:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr bool **value**

5.811.1 Detailed Description

```
template<typename _Tp>
struct std::is_final<_Tp>
```

`is_final`

Definition at line 715 of file `type_traits`.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.812 `std::is_floating_point< _Tp >` Struct Template Reference

Inherits `type< remove_cv< _Tp >::type >`.

5.812.1 Detailed Description

```
template<typename _Tp>
struct std::is_floating_point< _Tp >
```

`is_floating_point`

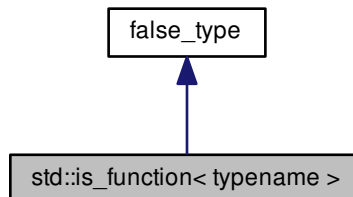
Definition at line 350 of file `type_traits`.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.813 `std::is_function< typename >` Struct Template Reference

Inheritance diagram for `std::is_function< typename >`:



Public Types

- typedef [integral_constant< _Tp, __v >](#) **type**
- typedef `_Tp` **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr `_Tp` **value**

5.813.1 Detailed Description

```
template<typename>
struct std::is_function< typename >
```

`is_function`

Definition at line 400 of file `type_traits`.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.814 `std::is_fundamental< _Tp >` Struct Template Reference

Inherits `type< is_arithmetic< _Tp >, is_void< _Tp >, is_null_pointer< _Tp > >`.

5.814.1 Detailed Description

```
template<typename _Tp>
struct std::is_fundamental< _Tp >
```

`is_fundamental`

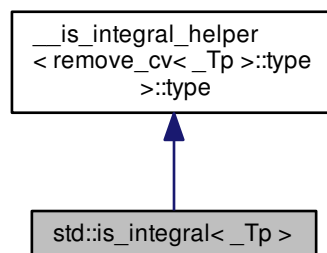
Definition at line 590 of file `type_traits`.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.815 `std::is_integral< _Tp >` Struct Template Reference

Inheritance diagram for `std::is_integral< _Tp >`:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr _Tp **value**

5.815.1 Detailed Description

```
template<typename _Tp>  
struct std::is_integral< _Tp >
```

is_integral

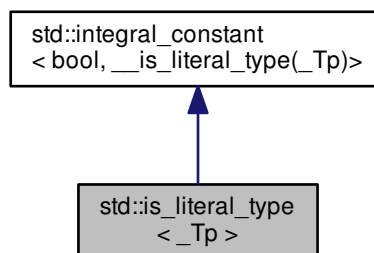
Definition at line 322 of file type_traits.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.816 std::is_literal_type< _Tp > Struct Template Reference

Inheritance diagram for std::is_literal_type< _Tp >:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr bool **value**

5.816.1 Detailed Description

```
template<typename _Tp>
struct std::is_literal_type< _Tp >
```

is_literal_type

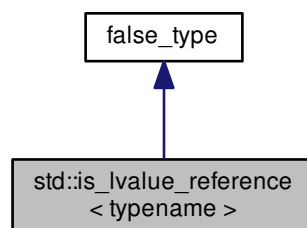
Definition at line 695 of file type_traits.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.817 std::is_lvalue_reference< typename > Struct Template Reference

Inheritance diagram for std::is_lvalue_reference< typename >:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr _Tp **value**

5.817.1 Detailed Description

```
template<typename>
struct std::is_lvalue_reference< typename >
```

is_lvalue_reference

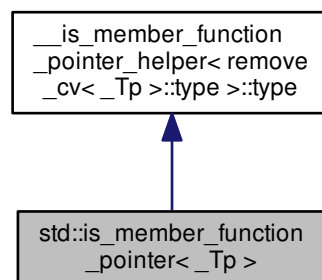
Definition at line 383 of file type_traits.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.818 std::is_member_function_pointer< _Tp > Struct Template Reference

Inheritance diagram for std::is_member_function_pointer< _Tp >:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr _Tp **value**

5.818.1 Detailed Description

```
template<typename _Tp>
struct std::is_member_function_pointer< _Tp >
```

is_member_function_pointer

Definition at line 427 of file type_traits.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.819 std::is_member_object_pointer< _Tp > Struct Template Reference

Inherits type< remove_cv< _Tp >::type >.

5.819.1 Detailed Description

```
template<typename _Tp>
struct std::is_member_object_pointer< _Tp >
```

is_member_object_pointer

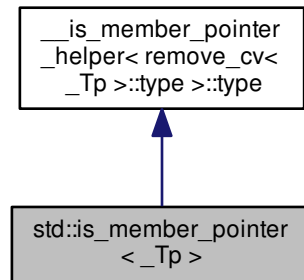
Definition at line 412 of file type_traits.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.820 std::is_member_pointer< _Tp > Struct Template Reference

Inheritance diagram for std::is_member_pointer< _Tp >:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr _Tp **value**

5.820.1 Detailed Description

```
template<typename _Tp>
struct std::is_member_pointer< _Tp >
```

is_member_pointer

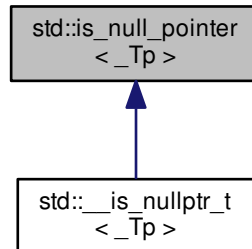
Definition at line 603 of file type_traits.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.821 `std::is_null_pointer<_Tp>` Struct Template Reference

Inheritance diagram for `std::is_null_pointer<_Tp>`:



5.821.1 Detailed Description

```
template<typename _Tp>
struct std::is_null_pointer<_Tp>
```

`is_null_pointer` (LWG 2247).

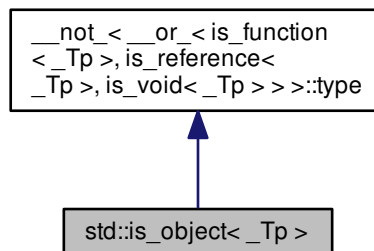
Definition at line 563 of file `type_traits`.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.822 `std::is_object<_Tp>` Struct Template Reference

Inheritance diagram for `std::is_object<_Tp>`:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr _Tp **value**

5.822.1 Detailed Description

```
template<typename _Tp>  
struct std::is_object< _Tp >
```

is_object

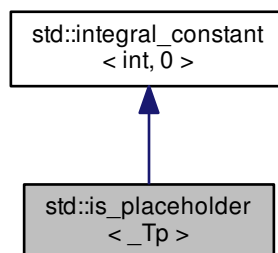
Definition at line 597 of file type_traits.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.823 std::is_placeholder< _Tp > Struct Template Reference

Inheritance diagram for std::is_placeholder< _Tp >:



Public Types

- typedef [integral_constant](#)< int, __v > **type**
- typedef int **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr int **value**

5.823.1 Detailed Description

```
template<typename _Tp>
struct std::is_placeholder< _Tp >
```

Determines if the given type `_Tp` is a placeholder in a `bind()` expression and, if so, which placeholder it is.

C++11 [func.bind.isplace].

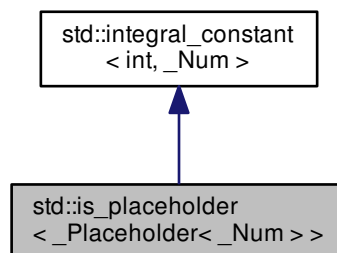
Definition at line 672 of file `functional`.

The documentation for this struct was generated from the following file:

- [functional](#)

5.824 `std::is_placeholder< _Placeholder< _Num > >` Struct Template Reference

Inheritance diagram for `std::is_placeholder< _Placeholder< _Num > >`:



Public Types

- typedef [integral_constant](#)< int, __v > **type**
- typedef int **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr int **value**

5.824.1 Detailed Description

```
template<int _Num>
struct std::is_placeholder< _Placeholder< _Num > >
```

Partial specialization of is_placeholder that provides the placeholder number for the placeholder objects defined by libstdc++.

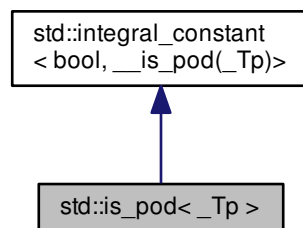
Definition at line 734 of file functional.

The documentation for this struct was generated from the following file:

- [functional](#)

5.825 std::is_pod< _Tp > Struct Template Reference

Inheritance diagram for std::is_pod< _Tp >:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr bool **value**

5.825.1 Detailed Description

```
template<typename _Tp>  
struct std::is_pod<_Tp>
```

is_pod

Definition at line 689 of file type_traits.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.826 std::is_pointer<_Tp> Struct Template Reference

Inherits type< remove_cv<_Tp>::type >.

5.826.1 Detailed Description

```
template<typename _Tp>  
struct std::is_pointer<_Tp>
```

is_pointer

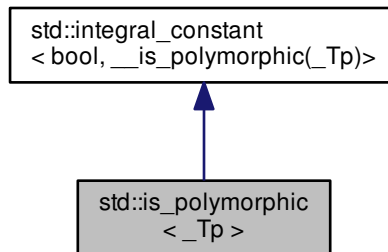
Definition at line 377 of file type_traits.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.827 `std::is_polymorphic<_Tp>` Struct Template Reference

Inheritance diagram for `std::is_polymorphic<_Tp>`:



Public Types

- typedef `integral_constant<bool, __v>` **type**
- typedef `bool` **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr `bool` **value**

5.827.1 Detailed Description

```
template<typename _Tp>  
struct std::is_polymorphic<_Tp>
```

`is_polymorphic`

Definition at line 707 of file `type_traits`.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.828 `std::is_reference< _Tp >` Struct Template Reference

Inherits type< `is_lvalue_reference< _Tp >`, `is_rvalue_reference< _Tp >` >.

5.828.1 Detailed Description

```
template<typename _Tp>
struct std::is_reference< _Tp >
```

`is_reference`

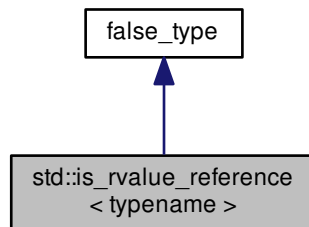
Definition at line 577 of file `type_traits`.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.829 `std::is_rvalue_reference< typename >` Struct Template Reference

Inheritance diagram for `std::is_rvalue_reference< typename >`:



Public Types

- typedef [integral_constant< _Tp, __v >](#) **type**
- typedef `_Tp` **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- `static constexpr _Tp value`

5.829.1 Detailed Description

```
template<typename>  
struct std::is_rvalue_reference< typename >
```

`is_rvalue_reference`

Definition at line 392 of file `type_traits`.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.830 `std::is_scalar< _Tp >` Struct Template Reference

Inherits type< `is_arithmetic< _Tp >`, `is_enum< _Tp >`, `is_pointer< _Tp >`, `is_member_pointer< _Tp >`, `is_null_pointer< _Tp >` >.

5.830.1 Detailed Description

```
template<typename _Tp>  
struct std::is_scalar< _Tp >
```

`is_scalar`

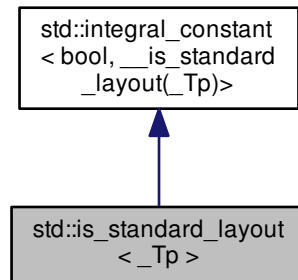
Definition at line 607 of file `type_traits`.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.831 `std::is_standard_layout<_Tp>` Struct Template Reference

Inheritance diagram for `std::is_standard_layout<_Tp>`:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr bool **value**

5.831.1 Detailed Description

```
template<typename _Tp>
struct std::is_standard_layout<_Tp>
```

`is_standard_layout`

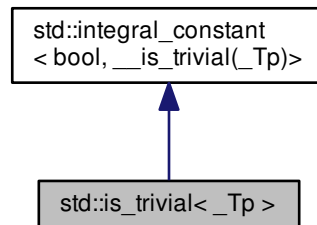
Definition at line 682 of file `type_traits`.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.832 std::is_trivial< _Tp > Struct Template Reference

Inheritance diagram for std::is_trivial< _Tp >:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr bool **value**

5.832.1 Detailed Description

```
template<typename _Tp>
struct std::is_trivial< _Tp >
```

is_trivial

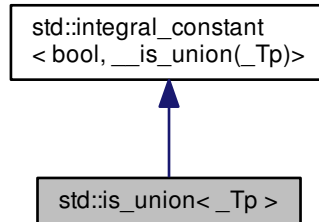
Definition at line 670 of file `type_traits`.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.833 std::is_union< _Tp > Struct Template Reference

Inheritance diagram for std::is_union< _Tp >:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr bool **value**

5.833.1 Detailed Description

```
template<typename _Tp>
struct std::is_union< _Tp >
```

is_union

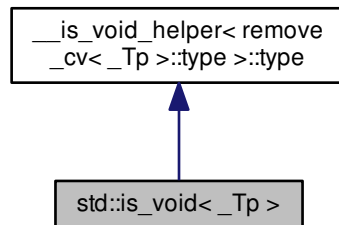
Definition at line 440 of file type_traits.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.834 std::is_void< _Tp > Struct Template Reference

Inheritance diagram for std::is_void< _Tp >:



Public Types

- typedef [integral_constant](#)< _Tp, __v > **type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr _Tp **value**

5.834.1 Detailed Description

```
template<typename _Tp>  
struct std::is_void< _Tp >
```

is_void

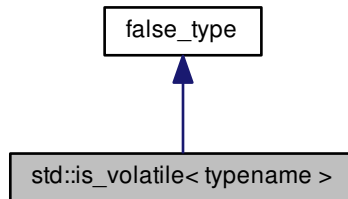
Definition at line 211 of file type_traits.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.835 `std::is_volatile< typename >` Struct Template Reference

Inheritance diagram for `std::is_volatile< typename >`:



Public Types

- typedef [integral_constant](#)< `_Tp`, `__v` > **type**
- typedef `_Tp` **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr `_Tp` **value**

5.835.1 Detailed Description

```
template<typename>
struct std::is_volatile< typename >
```

`is_volatile`

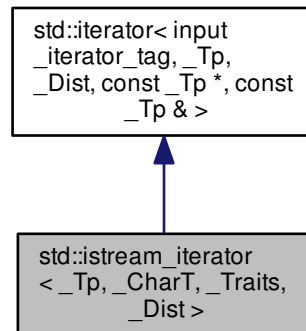
Definition at line 661 of file `type_traits`.

The documentation for this struct was generated from the following file:

- [type_traits](#)

5.836 std::istream_iterator< _Tp, _CharT, _Traits, _Dist > Class Template Reference

Inheritance diagram for std::istream_iterator< _Tp, _CharT, _Traits, _Dist >:



Public Types

- typedef _CharT **char_type**
- typedef _Dist **difference_type**
- typedef **basic_istream**< _CharT, _Traits > **istream_type**
- typedef **input_iterator_tag** **iterator_category**
- typedef const _Tp * **pointer**
- typedef const _Tp & **reference**
- typedef _Traits **traits_type**
- typedef _Tp **value_type**

Public Member Functions

- constexpr **istream_iterator** ()
- **istream_iterator** (**istream_type** &__s)
- **istream_iterator** (const **istream_iterator** &__obj)
- bool **_M_equal** (const **istream_iterator** &__x) const
- const _Tp & **operator*** () const
- **istream_iterator** & **operator++** ()
- **istream_iterator** **operator++** (int)
- const _Tp * **operator->** () const

5.836.1 Detailed Description

```

template<typename _Tp, typename _CharT = char, typename _Traits = char_traits<_CharT>, typename _Dist = ptrdiff_t>
class std::istream_iterator< _Tp, _CharT, _Traits, _Dist >

```

Provides input iterator semantics for streams.

Definition at line 49 of file stream_iterator.h.

5.836.2 Member Typedef Documentation

5.836.2.1 `typedef _Dist std::iterator< input_iterator_tag , _Tp, _Dist , const _Tp * , const _Tp & >::difference_type`
[inherited]

Distance between iterators is represented as this type.

Definition at line 125 of file `stl_iterator_base_types.h`.

5.836.2.2 `typedef input_iterator_tag std::iterator< input_iterator_tag , _Tp, _Dist , const _Tp * , const _Tp & >::iterator_category` [inherited]

One of the [tag types](#).

Definition at line 121 of file `stl_iterator_base_types.h`.

5.836.2.3 `typedef const _Tp * std::iterator< input_iterator_tag , _Tp, _Dist , const _Tp * , const _Tp & >::pointer`
[inherited]

This type represents a pointer-to-value_type.

Definition at line 127 of file `stl_iterator_base_types.h`.

5.836.2.4 `typedef const _Tp & std::iterator< input_iterator_tag , _Tp, _Dist , const _Tp * , const _Tp & >::reference`
[inherited]

This type represents a reference-to-value_type.

Definition at line 129 of file `stl_iterator_base_types.h`.

5.836.2.5 `typedef _Tp std::iterator< input_iterator_tag , _Tp, _Dist , const _Tp * , const _Tp & >::value_type`
[inherited]

The type "pointed to" by the iterator.

Definition at line 123 of file `stl_iterator_base_types.h`.

5.836.3 Constructor & Destructor Documentation

5.836.3.1 `template<typename _Tp, typename _CharT = char, typename _Traits = char_traits<_CharT>, typename _Dist = ptrdiff_t>`
`constexpr std::istream_iterator< _Tp, _CharT, _Traits, _Dist >::istream_iterator ()` [inline]

Construct end of input stream iterator.

Definition at line 64 of file `stream_iterator.h`.

Referenced by `std::istream_iterator< _Tp, _CharT, _Traits, _Dist >::istream_iterator()`.

5.836.3.2 `template<typename _Tp, typename _CharT = char, typename _Traits = char_traits<_CharT>, typename _Dist = ptrdiff_t>
std::istream_iterator<_Tp, _CharT, _Traits, _Dist>::istream_iterator(istream_type & __s) [inline]`

Construct start of input stream iterator.

Definition at line 68 of file `stream_iterator.h`.

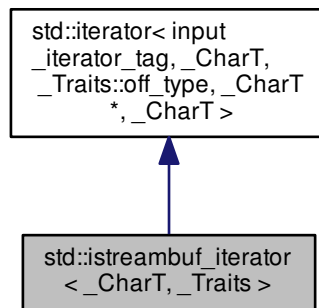
References `std::istream_iterator<_Tp, _CharT, _Traits, _Dist>::istream_iterator()`.

The documentation for this class was generated from the following file:

- [stream_iterator.h](#)

5.837 std::istreambuf_iterator<_CharT, _Traits> Class Template Reference

Inheritance diagram for `std::istreambuf_iterator<_CharT, _Traits>`:



Public Types

- `typedef _Traits::off_type` [difference_type](#)
- `typedef` [input_iterator_tag](#) `iterator_category`
- `typedef _CharT *` [pointer](#)
- `typedef _CharT` [reference](#)
- `typedef _CharT` [value_type](#)
- `typedef _CharT` [char_type](#)
- `typedef _Traits` [traits_type](#)
- `typedef _Traits::int_type` [int_type](#)
- `typedef` [basic_streambuf<_CharT, _Traits>](#) [streambuf_type](#)
- `typedef` [basic_istream<_CharT, _Traits>](#) [istream_type](#)

Public Member Functions

- constexpr [istreambuf_iterator](#) () noexcept
- **istreambuf_iterator** (const [istreambuf_iterator](#) &) noexcept=default
- [istreambuf_iterator](#) (istream_type & __s) noexcept
- [istreambuf_iterator](#) (streambuf_type * __s) noexcept
- bool [equal](#) (const [istreambuf_iterator](#) & __b) const
- [char_type](#) [operator*](#) () const
- [istreambuf_iterator](#) & [operator++](#) ()
- [istreambuf_iterator](#) [operator++](#) (int)

Friends

- template<bool _IsMove, typename _CharT2 >
__gnu_cxx::__enable_if< __is_char< _CharT2 >::__value, _CharT2 * >::__type **__copy_move_a2**
([istreambuf_iterator](#)< _CharT2 >, [istreambuf_iterator](#)< _CharT2 >, _CharT2 *)
- template<typename _CharT2 >
__gnu_cxx::__enable_if< __is_char< _CharT2 >::__value, [ostreambuf_iterator](#)< _CharT2 >::__type **copy**
([istreambuf_iterator](#)< _CharT2 >, [istreambuf_iterator](#)< _CharT2 >, [ostreambuf_iterator](#)< _CharT2 >)
- template<typename _CharT2 >
__gnu_cxx::__enable_if< __is_char< _CharT2 >::__value, [istreambuf_iterator](#)< _CharT2 >::__type **find**
([istreambuf_iterator](#)< _CharT2 >, [istreambuf_iterator](#)< _CharT2 >, const _CharT2 &)

5.837.1 Detailed Description

```
template<typename _CharT, typename _Traits>
class std::istreambuf_iterator< _CharT, _Traits >
```

Provides input iterator semantics for streambufs.

Definition at line 125 of file iosfwd.

5.837.2 Member Typedef Documentation

5.837.2.1 `template<typename _CharT , typename _Traits > typedef _CharT std::istreambuf_iterator< _CharT, _Traits >::char_type`

Public typedefs.

Definition at line 64 of file streambuf_iterator.h.

5.837.2.2 `typedef _Traits::off_type std::iterator< input_iterator_tag , _CharT , _Traits::off_type , _CharT * , _CharT >::difference_type` [inherited]

Distance between iterators is represented as this type.

Definition at line 125 of file stl_iterator_base_types.h.

5.837.2.3 `template<typename _CharT, typename _Traits> typedef _Traits::int_type std::istreambuf_iterator<_CharT, _Traits>::int_type`

Public typedefs.

Definition at line 66 of file `streambuf_iterator.h`.

5.837.2.4 `template<typename _CharT, typename _Traits> typedef basic_istream<_CharT, _Traits> std::istreambuf_iterator<_CharT, _Traits>::istream_type`

Public typedefs.

Definition at line 68 of file `streambuf_iterator.h`.

5.837.2.5 `typedef input_iterator_tag std::iterator< input_iterator_tag, _CharT, _Traits::off_type, _CharT*, _CharT>::iterator_category` `[inherited]`

One of the [tag types](#).

Definition at line 121 of file `stl_iterator_base_types.h`.

5.837.2.6 `typedef _CharT* std::iterator< input_iterator_tag, _CharT, _Traits::off_type, _CharT*, _CharT>::pointer` `[inherited]`

This type represents a `pointer-to-value_type`.

Definition at line 127 of file `stl_iterator_base_types.h`.

5.837.2.7 `typedef _CharT std::iterator< input_iterator_tag, _CharT, _Traits::off_type, _CharT*, _CharT>::reference` `[inherited]`

This type represents a `reference-to-value_type`.

Definition at line 129 of file `stl_iterator_base_types.h`.

5.837.2.8 `template<typename _CharT, typename _Traits> typedef basic_streambuf<_CharT, _Traits> std::istreambuf_iterator<_CharT, _Traits>::streambuf_type`

Public typedefs.

Definition at line 67 of file `streambuf_iterator.h`.

5.837.2.9 `template<typename _CharT, typename _Traits> typedef _Traits std::istreambuf_iterator<_CharT, _Traits>::traits_type`

Public typedefs.

Definition at line 65 of file `streambuf_iterator.h`.

5.837.2.10 `typedef _CharT std::iterator< input_iterator_tag , _CharT , _Traits::off_type , _CharT * , _CharT >::value_type`
`[inherited]`

The type "pointed to" by the iterator.

Definition at line 123 of file `stl_iterator_base_types.h`.

5.837.3 Constructor & Destructor Documentation

5.837.3.1 `template<typename _CharT , typename _Traits > constexpr std::istreambuf_iterator< _CharT, _Traits`
`>::istreambuf_iterator () [inline], [noexcept]`

Construct end of input stream iterator.

Definition at line 102 of file `streambuf_iterator.h`.

5.837.3.2 `template<typename _CharT , typename _Traits > std::istreambuf_iterator< _CharT, _Traits`
`>::istreambuf_iterator (istream_type & __s) [inline], [noexcept]`

Construct start of input stream iterator.

Definition at line 112 of file `streambuf_iterator.h`.

5.837.3.3 `template<typename _CharT , typename _Traits > std::istreambuf_iterator< _CharT, _Traits`
`>::istreambuf_iterator (streambuf_type * __s) [inline], [noexcept]`

Construct start of streambuf iterator.

Definition at line 116 of file `streambuf_iterator.h`.

5.837.4 Member Function Documentation

5.837.4.1 `template<typename _CharT , typename _Traits > bool std::istreambuf_iterator< _CharT, _Traits >::equal (const`
`istreambuf_iterator< _CharT, _Traits > & __b) const [inline]`

Return true both iterators are end or both are not end.

Definition at line 172 of file `streambuf_iterator.h`.

References `std::basic_streambuf< _CharT, _Traits >::sgetc()`.

5.837.4.2 `template<typename _CharT , typename _Traits > char_type std::istreambuf_iterator< _CharT, _Traits`
`>::operator* () const [inline]`

Return the current character pointed to by iterator. This returns `streambuf.sgetc()`. It cannot be assigned. NB: The result of `operator*()` on an end of stream is undefined.

Definition at line 123 of file `streambuf_iterator.h`.

5.837.4.3 `template<typename _CharT, typename _Traits > istreambuf_iterator& std::istreambuf_iterator< _CharT, _Traits >::operator++() [inline]`

Advance the iterator. Calls `streambuf.sbumpc()`.

Definition at line 137 of file `streambuf_iterator.h`.

References `std::basic_streambuf< _CharT, _Traits >::sbumpc()`.

5.837.4.4 `template<typename _CharT, typename _Traits > istreambuf_iterator std::istreambuf_iterator< _CharT, _Traits >::operator++(int) [inline]`

Advance the iterator. Calls `streambuf.sbumpc()`.

Definition at line 152 of file `streambuf_iterator.h`.

References `std::basic_streambuf< _CharT, _Traits >::sbumpc()`.

The documentation for this class was generated from the following files:

- [iosfwd](#)
- [streambuf_iterator.h](#)

5.838 `std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >` Struct Template Reference

Public Types

- typedef `_Distance` [difference_type](#)
- typedef `_Category` [iterator_category](#)
- typedef `_Pointer` [pointer](#)
- typedef `_Reference` [reference](#)
- typedef `_Tp` [value_type](#)

5.838.1 Detailed Description

```
template<typename _Category, typename _Tp, typename _Distance = ptrdiff_t, typename _Pointer = _Tp*, typename _Reference =
_Tp&>
struct std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >
```

Common iterator class.

This class does nothing but define nested typedefs. Iterator classes can inherit from this class to save some work. The typedefs are then used in specializations and overloading.

In particular, there are no default implementations of requirements such as `operator++` and the like. (How could there be?)

Definition at line 118 of file `stl_iterator_base_types.h`.

5.838.2 Member Typedef Documentation

5.838.2.1 `template<typename _Category, typename _Tp, typename _Distance = ptrdiff_t, typename _Pointer = _Tp*,
typename _Reference = _Tp&> typedef _Distance std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference
>::difference_type`

Distance between iterators is represented as this type.

Definition at line 125 of file `stl_iterator_base_types.h`.

5.838.2.2 `template<typename _Category, typename _Tp, typename _Distance = ptrdiff_t, typename _Pointer = _Tp*,
typename _Reference = _Tp&> typedef _Category std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference
>::iterator_category`

One of the [tag types](#).

Definition at line 121 of file `stl_iterator_base_types.h`.

5.838.2.3 `template<typename _Category, typename _Tp, typename _Distance = ptrdiff_t, typename _Pointer = _Tp*, typename
_Reference = _Tp&> typedef _Pointer std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >::pointer`

This type represents a pointer-to-value_type.

Definition at line 127 of file `stl_iterator_base_types.h`.

5.838.2.4 `template<typename _Category, typename _Tp, typename _Distance = ptrdiff_t, typename _Pointer = _Tp*, typename
_Reference = _Tp&> typedef _Reference std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >::reference`

This type represents a reference-to-value_type.

Definition at line 129 of file `stl_iterator_base_types.h`.

5.838.2.5 `template<typename _Category, typename _Tp, typename _Distance = ptrdiff_t, typename _Pointer = _Tp*, typename
_Reference = _Tp&> typedef _Tp std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >::value_type`

The type "pointed to" by the iterator.

Definition at line 123 of file `stl_iterator_base_types.h`.

The documentation for this struct was generated from the following file:

- [stl_iterator_base_types.h](#)

5.839 std::iterator_traits< _Tp * > Struct Template Reference

Public Types

- typedef `ptrdiff_t` **difference_type**
- typedef [random_access_iterator_tag](#) **iterator_category**
- typedef `_Tp *` **pointer**
- typedef `_Tp &` **reference**
- typedef `_Tp` **value_type**

5.839.1 Detailed Description

```
template<typename _Tp>
struct std::iterator_traits< _Tp * >
```

Partial specialization for pointer types.

Definition at line 178 of file `stl_iterator_base_types.h`.

The documentation for this struct was generated from the following file:

- [stl_iterator_base_types.h](#)

5.840 `std::iterator_traits< const _Tp * >` Struct Template Reference

Public Types

- typedef ptrdiff_t **difference_type**
- typedef [random_access_iterator_tag](#) **iterator_category**
- typedef const _Tp * **pointer**
- typedef const _Tp & **reference**
- typedef _Tp **value_type**

5.840.1 Detailed Description

```
template<typename _Tp>
struct std::iterator_traits< const _Tp * >
```

Partial specialization for const pointer types.

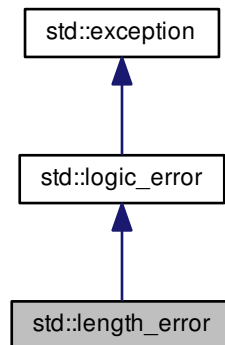
Definition at line 189 of file `stl_iterator_base_types.h`.

The documentation for this struct was generated from the following file:

- [stl_iterator_base_types.h](#)

5.841 `std::length_error` Class Reference

Inheritance diagram for `std::length_error`:



Public Member Functions

- **`length_error`** (const [string](#) &__arg) `_GLIBCXX_TXN_SAFE`
- **`length_error`** (const char *) `_GLIBCXX_TXN_SAFE`
- virtual const char * [what](#) () const `_GLIBCXX_TXN_SAFE_DYN noexcept`

5.841.1 Detailed Description

Thrown when an object is constructed that would exceed its maximum permitted size (e.g., a `basic_string` instance).

Definition at line 170 of file `stdexcept`.

5.841.2 Member Function Documentation

5.841.2.1 `virtual const char* std::length_error::what () const` [`virtual`], [`noexcept`], [`inherited`]

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

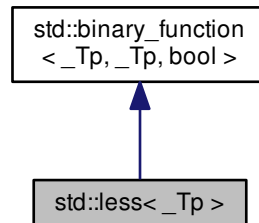
Reimplemented in [std::future_error](#).

The documentation for this class was generated from the following file:

- [stdexcept](#)

5.842 std::less< _Tp > Struct Template Reference

Inheritance diagram for std::less< _Tp >:



Public Types

- typedef `_Tp` [first_argument_type](#)
- typedef `bool` [result_type](#)
- typedef `_Tp` [second_argument_type](#)

Public Member Functions

- `_GLIBCXX14_CONSTEXPR bool operator() (const _Tp &__x, const _Tp &__y) const`

5.842.1 Detailed Description

```
template<typename _Tp>
struct std::less< _Tp >
```

One of the [comparison functors](#).

Definition at line 340 of file `stl_function.h`.

5.842.2 Member Typedef Documentation

5.842.2.1 typedef `_Tp` `std::binary_function< _Tp, _Tp, bool >::first_argument_type` [\[inherited\]](#)

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.842.2.2 `typedef bool std::binary_function<_Tp, _Tp, bool>::result_type` [inherited]

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.842.2.3 `typedef _Tp std::binary_function<_Tp, _Tp, bool>::second_argument_type` [inherited]

`second_argument_type` is the type of the second argument

Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.843 `std::less< void >` Struct Template Reference

Public Types

- `typedef __is_transparent is_transparent`

Public Member Functions

- `template<typename _Tp, typename _Up >
_GLIBCXX14_CONSTEXPR auto operator() (_Tp &&__t, _Up &&__u) const noexcept(noexcept(std::forward<
_Tp >(__t)< std::forward< _Up >(__u))) -> decltype(std::forward< _Tp >(__t)< std::forward< _Up >(__u))`

5.843.1 Detailed Description

```
template<>
struct std::less< void >
```

One of the [comparison functors](#).

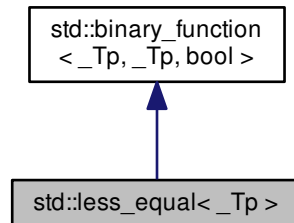
Definition at line 457 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.844 std::less_equal<_Tp> Struct Template Reference

Inheritance diagram for std::less_equal<_Tp>:



Public Types

- typedef _Tp [first_argument_type](#)
- typedef bool [result_type](#)
- typedef _Tp [second_argument_type](#)

Public Member Functions

- _GLIBCXX14_CONSTEXPR bool **operator()** (const _Tp &__x, const _Tp &__y) const

5.844.1 Detailed Description

```
template<typename _Tp>
struct std::less_equal<_Tp>
```

One of the [comparison functors](#).

Definition at line 346 of file stl_function.h.

5.844.2 Member Typedef Documentation

5.844.2.1 typedef _Tp std::binary_function<_Tp, _Tp, bool>::**first_argument_type** [inherited]

`first_argument_type` is the type of the first argument

Definition at line 121 of file stl_function.h.

5.844.2.2 `typedef bool std::binary_function<_Tp, _Tp, bool>::result_type` [inherited]

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.844.2.3 `typedef _Tp std::binary_function<_Tp, _Tp, bool>::second_argument_type` [inherited]

`second_argument_type` is the type of the second argument

Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.845 `std::less_equal< void >` Struct Template Reference

Public Types

- `typedef __is_transparent is_transparent`

Public Member Functions

- `template<typename _Tp, typename _Up >
_GLIBCXX14_CONSTEXPR auto operator() (_Tp &&__t, _Up &&__u) const noexcept(noexcept(std::forward<
_Tp >(__t)<=std::forward< _Up >(__u))) -> decltype(std::forward< _Tp >(__t)<=std::forward< _Up >(__u))`

5.845.1 Detailed Description

```
template<>
struct std::less_equal< void >
```

One of the [comparison functors](#).

Definition at line 487 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.846 `std::linear_congruential_engine< _UIntType, __a, __c, __m >` Class Template Reference

Public Types

- `typedef _UIntType result_type`

Public Member Functions

- [linear_congruential_engine](#) ([result_type](#) __s=default_seed)
- [template](#)<typename _Sseq, typename = typename std::enable_if<!std::is_same<_Sseq, linear_congruential_engine>::value>::type> [linear_congruential_engine](#) (_Sseq &__q)
- void [discard](#) (unsigned long long __z)
- [result_type](#) [operator](#)() ()
- void [seed](#) ([result_type](#) __s=default_seed)
- [template](#)<typename _Sseq > std::enable_if< [std::is_class](#)<_Sseq>::value >::type [seed](#) (_Sseq &__q)

Static Public Member Functions

- static constexpr [result_type](#) [max](#) ()
- static constexpr [result_type](#) [min](#) ()

Static Public Attributes

- static constexpr [result_type](#) [default_seed](#)
- static constexpr [result_type](#) [increment](#)
- static constexpr [result_type](#) [modulus](#)
- static constexpr [result_type](#) [multiplier](#)

Friends

- [template](#)<typename _UIntType1, _UIntType1 __a1, _UIntType1 __c1, _UIntType1 __m1, typename _CharT, typename _Traits > [std::basic_ostream](#)<_CharT, _Traits > & [operator<<](#) ([std::basic_ostream](#)<_CharT, _Traits > &__os, const [std::linear_congruential_engine](#)<_UIntType1, __a1, __c1, __m1 > &__lcr)
- bool [operator==](#) (const [linear_congruential_engine](#) &__lhs, const [linear_congruential_engine](#) &__rhs)
- [template](#)<typename _UIntType1, _UIntType1 __a1, _UIntType1 __c1, _UIntType1 __m1, typename _CharT, typename _Traits > [std::basic_istream](#)<_CharT, _Traits > & [operator>>](#) ([std::basic_istream](#)<_CharT, _Traits > &__is, [std::linear_congruential_engine](#)<_UIntType1, __a1, __c1, __m1 > &__lcr)

5.846.1 Detailed Description

```
template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m>
class std::linear_congruential_engine<_UIntType, __a, __c, __m>
```

A model of a linear congruential random number generator.

A random number generator that produces pseudorandom numbers via linear function:

$$x_{i+1} \leftarrow (ax_i + c) \bmod m$$

The template parameter `_UIntType` must be an unsigned integral type large enough to store values up to `(__m-1)`. If the template parameter `__m` is 0, the modulus `__m` used is `std::numeric_limits<_UIntType>::max()` plus 1. Otherwise, the template parameters `__a` and `__c` must be less than `__m`.

The size of the state is 1.

Definition at line 236 of file random.h.

5.846.2 Member Typedef Documentation

5.846.2.1 `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m> typedef _UIntType
std::linear_congruential_engine< _UIntType, __a, __c, __m >::result_type`

The type of the generated random value.

Definition at line 239 of file random.h.

5.846.3 Constructor & Destructor Documentation

5.846.3.1 `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m> std::linear_congruential_engine<
_UIntType, __a, __c, __m >::linear_congruential_engine (result_type __s = default_seed)
[inline], [explicit]`

Constructs a linear_congruential_engine random number generator engine with seed __s. The default seed value is 1.

Parameters

<code>__s</code>	The initial seed value.
------------------	-------------------------

Definition at line 263 of file random.h.

5.846.3.2 `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m> template<typename _Sseq ,
typename = typename std::enable_if<!std::is_same<_Sseq, linear_congruential_engine>::value> ::type>
std::linear_congruential_engine< _UIntType, __a, __c, __m >::linear_congruential_engine (_Sseq & __q)
[inline], [explicit]`

Constructs a linear_congruential_engine random number generator engine seeded from the seed sequence __q.

Parameters

<code>__q</code>	the seed sequence.
------------------	--------------------

Definition at line 276 of file random.h.

5.846.4 Member Function Documentation

5.846.4.1 `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m> void std::linear_↵
_congruential_engine< _UIntType, __a, __c, __m >::discard (unsigned long long __z)
[inline]`

Discard a sequence of random numbers.

Definition at line 320 of file random.h.

5.846.4.2 `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m> static constexpr result_type
std::linear_congruential_engine<_UIntType, __a, __c, __m>::max () [inline], [static]`

Gets the largest possible value in the output range.

Definition at line 313 of file random.h.

5.846.4.3 `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m> static constexpr result_type
std::linear_congruential_engine<_UIntType, __a, __c, __m>::min () [inline], [static]`

Gets the smallest possible value in the output range.

The minimum depends on the __c parameter: if it is zero, the minimum generated must be > 0, otherwise 0 is allowed.

Definition at line 306 of file random.h.

5.846.4.4 `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m> result_type
std::linear_congruential_engine<_UIntType, __a, __c, __m>::operator() () [inline]`

Gets the next random number in the sequence.

Definition at line 330 of file random.h.

5.846.4.5 `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m> void std::linear_↵
congruential_engine<_UIntType, __a, __c, __m>::seed (result_type __s = default_seed
)`

Reseeds the linear_congruential_engine random number generator engine sequence to the seed __s.

Parameters

<code>↵ __s</code>	The new seed.
------------------------	---------------

Seeds the LCR with integral value __s, adjusted so that the ring identity is never a member of the convergence set.

Definition at line 120 of file bits/random.tcc.

5.846.4.6 `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m> template<typename _Sseq >
std::enable_if< std::is_class<_Sseq>::value >::type std::linear_congruential_engine<_UIntType, __a, __c,
__m>::seed (_Sseq & __q)`

Reseeds the linear_congruential_engine random number generator engine sequence using values from the seed sequence __q.

Parameters

<code>↵ __q</code>	the seed sequence.
------------------------	--------------------

Seeds the LCR engine with a value generated by `__q`.

Definition at line 136 of file `bits/random.tcc`.

References `std::__lg()`, `std::dec()`, `std::fixed()`, `std::ios_base::flags()`, `std::left()`, and `std::__detail::operator>>()`.

5.846.5 Friends And Related Function Documentation

5.846.5.1 `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m> template<typename _UIntType1, _UIntType1 __a1, _UIntType1 __c1, _UIntType1 __m1, typename _CharT, typename _Traits > std::basic_ostream<_CharT, _Traits>& operator<< (std::basic_ostream<_CharT, _Traits> & __os, const std::linear_congruential_engine<_UIntType1, __a1, __c1, __m1> & __lcr) [friend]`

Writes the textual representation of the state `x(i)` of `x` to `__os`.

Parameters

<code>__os</code>	The output stream.
<code>__lcr</code>	A % <code>linear_congruential_engine</code> random number generator.

Returns

`__os`.

5.846.5.2 `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m> bool operator== (const linear_congruential_engine<_UIntType, __a, __c, __m> & __lhs, const linear_congruential_engine<_UIntType, __a, __c, __m> & __rhs) [friend]`

Compares two linear congruential random number generator objects of the same type for equality.

Parameters

<code>__lhs</code>	A linear congruential random number generator object.
<code>__rhs</code>	Another linear congruential random number generator object.

Returns

`true` if the infinite sequences of generated values would be equal, `false` otherwise.

Definition at line 348 of file `random.h`.

5.846.5.3 `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m> template<typename _UIntType1, _UIntType1 __a1, _UIntType1 __c1, _UIntType1 __m1, typename _CharT, typename _Traits > std::basic_istream<_CharT, _Traits>& operator>> (std::basic_istream<_CharT, _Traits> & __is, std::linear_congruential_engine<_UIntType1, __a1, __c1, __m1> & __lcr) [friend]`

Sets the state of the engine by reading its textual representation from `__is`.

The textual representation must have been previously written using an output stream whose imbued locale and whose type's template specialization arguments `_CharT` and `_Traits` were the same as those of `__is`.

Parameters

<code>__is</code>	The input stream.
<code>__lcr</code>	A % linear_congruential_engine random number generator.

Returns

`__is`.

5.846.6 Member Data Documentation

5.846.6.1 `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m> constexpr _UIntType
std::linear_congruential_engine<_UIntType, __a, __c, __m>::increment [static]`

An increment.

Definition at line 250 of file random.h.

5.846.6.2 `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m> constexpr _UIntType
std::linear_congruential_engine<_UIntType, __a, __c, __m>::modulus [static]`

The modulus.

Definition at line 252 of file random.h.

5.846.6.3 `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m> constexpr _UIntType
std::linear_congruential_engine<_UIntType, __a, __c, __m>::multiplier [static]`

The multiplier.

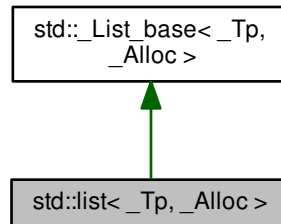
Definition at line 248 of file random.h.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.847 `std::list<_Tp, _Alloc>` Class Template Reference

Inheritance diagram for `std::list<_Tp, _Alloc>`:



Public Types

- `typedef _Alloc allocator_type`
- `typedef _List_const_iterator<_Tp> const_iterator`
- `typedef _Tp_alloc_traits::const_pointer const_pointer`
- `typedef _Tp_alloc_traits::const_reference const_reference`
- `typedef std::reverse_iterator< const_iterator > const_reverse_iterator`
- `typedef ptrdiff_t difference_type`
- `typedef _List_iterator<_Tp> iterator`
- `typedef _Tp_alloc_traits::pointer pointer`
- `typedef _Tp_alloc_traits::reference reference`
- `typedef std::reverse_iterator< iterator > reverse_iterator`
- `typedef size_t size_type`
- `typedef _Tp value_type`

Public Member Functions

- `list()` `noexcept(is_nothrow_default_constructible<_Node_alloc_type>::value)`
- `list(const allocator_type &__a)` `noexcept`
- `list(size_type __n, const allocator_type &__a=allocator_type())`
- `list(size_type __n, const value_type &__value, const allocator_type &__a=allocator_type())`
- `list(const list &__x)`
- `list(list &&__x)` `noexcept`
- `list(initializer_list< value_type > __l, const allocator_type &__a=allocator_type())`
- `list(const list &__x, const allocator_type &__a)`
- `list(list &&__x, const allocator_type &__a)` `noexcept(_Node_alloc_traits::_S_always_equal())`
- `template<typename _InputIterator, typename = std::RequireInputIter<_InputIterator>>>`
`list(_InputIterator __first, _InputIterator __last, const allocator_type &__a=allocator_type())`
- `void assign(size_type __n, const value_type &__val)`

- template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>>
void **assign** (_InputIterator __first, _InputIterator __last)
- void **assign** (initializer_list< value_type > __l)
- reference **back** () noexcept
- const_reference **back** () const noexcept
- **iterator begin** () noexcept
- **const_iterator begin** () const noexcept
- **const_iterator cbegin** () const noexcept
- **const_iterator cend** () const noexcept
- void **clear** () noexcept
- **const_reverse_iterator crbegin** () const noexcept
- **const_reverse_iterator crend** () const noexcept
- template<typename... _Args>
iterator emplace (const_iterator __position, _Args &&...__args)
- template<typename... _Args>
void **emplace_back** (_Args &&...__args)
- template<typename... _Args>
void **emplace_front** (_Args &&...__args)
- bool **empty** () const noexcept
- **iterator end** () noexcept
- **const_iterator end** () const noexcept
- **iterator erase** (const_iterator __position) noexcept
- **iterator erase** (const_iterator __first, const_iterator __last) noexcept
- reference **front** () noexcept
- const_reference **front** () const noexcept
- allocator_type **get_allocator** () const noexcept
- **iterator insert** (const_iterator __position, const value_type &__x)
- **iterator insert** (const_iterator __position, value_type &&__x)
- **iterator insert** (const_iterator __p, initializer_list< value_type > __l)
- **iterator insert** (const_iterator __position, size_type __n, const value_type &__x)
- template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>>
iterator insert (const_iterator __position, _InputIterator __first, _InputIterator __last)
- size_type **max_size** () const noexcept
- void **merge** (list &&__x)
- void **merge** (list &__x)
- template<typename _StrictWeakOrdering >
void **merge** (list &&__x, _StrictWeakOrdering __comp)
- template<typename _StrictWeakOrdering >
void **merge** (list &__x, _StrictWeakOrdering __comp)
- list & **operator=** (const list &__x)
- list & **operator=** (list &&__x) noexcept(_Node_alloc_traits::_S_nothrow_move())
- list & **operator=** (initializer_list< value_type > __l)
- void **pop_back** () noexcept
- void **pop_front** () noexcept
- void **push_back** (const value_type &__x)
- void **push_back** (value_type &&__x)
- void **push_front** (const value_type &__x)
- void **push_front** (value_type &&__x)
- **reverse_iterator rbegin** () noexcept
- **const_reverse_iterator rbegin** () const noexcept
- void **remove** (const _Tp &__value)

- `template<typename _Predicate >`
`void remove_if (_Predicate)`
- `reverse_iterator rend () noexcept`
- `const_reverse_iterator rend () const noexcept`
- `void resize (size_type __new_size)`
- `void resize (size_type __new_size, const value_type &__x)`
- `void reverse () noexcept`
- `size_type size () const noexcept`
- `void sort ()`
- `template<typename _StrictWeakOrdering >`
`void sort (_StrictWeakOrdering)`
- `void splice (const_iterator __position, list &&__x) noexcept`
- `void splice (const_iterator __position, list &__x) noexcept`
- `void splice (const_iterator __position, list &&__x, const_iterator __i) noexcept`
- `void splice (const_iterator __position, list &__x, const_iterator __i) noexcept`
- `void splice (const_iterator __position, list &&__x, const_iterator __first, const_iterator __last) noexcept`
- `void splice (const_iterator __position, list &__x, const_iterator __first, const_iterator __last) noexcept`
- `void swap (list &__x) noexcept`
- `void unique ()`
- `template<typename _BinaryPredicate >`
`void unique (_BinaryPredicate)`

Protected Types

- `typedef _List_node< _Tp > _Node`

Protected Member Functions

- `template<typename _Integer >`
`void _M_assign_dispatch (_Integer __n, _Integer __val, __true_type)`
- `template<typename _InputIterator >`
`void _M_assign_dispatch (_InputIterator __first, _InputIterator __last, __false_type)`
- `void _M_check_equal_allocators (list &__x) noexcept`
- `void _M_clear () noexcept`
- `template<typename... _Args>`
`_Node * _M_create_node (_Args &&... __args)`
- `void _M_dec_size (size_t)`
- `void _M_default_append (size_type __n)`
- `void _M_default_initialize (size_type __n)`
- `size_t _M_distance (const void *, const void *) const`
- `void _M_erase (iterator __position) noexcept`
- `void _M_fill_assign (size_type __n, const value_type &__val)`
- `void _M_fill_initialize (size_type __n, const value_type &__x)`
- `_Node_alloc_traits::pointer _M_get_node ()`
- `_Node_alloc_type & _M_get_Node_allocator () noexcept`
- `const _Node_alloc_type & _M_get_Node_allocator () const noexcept`
- `size_t _M_get_size () const`
- `void _M_inc_size (size_t)`
- `void _M_init () noexcept`

- `template<typename _Integer>`
`void _M_initialize_dispatch (_Integer __n, _Integer __x, __true_type)`
- `template<typename _InputIterator>`
`void _M_initialize_dispatch (_InputIterator __first, _InputIterator __last, __false_type)`
- `template<typename... _Args>`
`void _M_insert (iterator __position, _Args &&... __args)`
- `void _M_move_assign (list &&__x, true_type) noexcept`
- `void _M_move_assign (list &&__x, false_type)`
- `void _M_move_nodes (_List_base &&__x)`
- `size_t _M_node_count () const`
- `void _M_put_node (typename _Node_alloc_traits::pointer __p) noexcept`
- `const_iterator _M_resize_pos (size_type &__new_size) const`
- `void _M_set_size (size_t)`
- `void _M_transfer (iterator __position, iterator __first, iterator __last)`

Static Protected Member Functions

- `static size_t _S_distance (const __detail::List_node_base * __first, const __detail::List_node_base * __last)`

Protected Attributes

- `_List_impl _M_impl`

5.847.1 Detailed Description

```
template<typename _Tp, typename _Alloc = std::allocator<_Tp>>
class std::list<_Tp, _Alloc>
```

A standard container with linear time access to elements, and fixed time insertion/deletion at any point in the sequence.

Template Parameters

<code>_Tp</code>	Type of element.
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_Tp></code> .

Meets the requirements of a [container](#), a [reversible container](#), and a [sequence](#), including the [optional sequence requirements](#) with the exception of `at` and `operator[]`.

This is a *doubly linked* list. Traversal up and down the list requires linear time, but adding and removing elements (or *nodes*) is done in constant time, regardless of where the change takes place. Unlike `std::vector` and `std::deque`, random-access iterators are not provided, so subscripting (`[]`) access is not allowed. For algorithms which only need sequential access, this lack makes no difference.

Also unlike the other standard containers, `std::list` provides specialized algorithms unique to linked lists, such as splicing, sorting, and in-place reversal.

A couple points on memory allocation for `list<Tp>`:

First, we never actually allocate a `Tp`, we allocate `List_node<Tp>`'s and trust [20.1.5]/4 to DTRT. This is to ensure that after elements from `list<X,Alloc1>` are spliced into `list<X,Alloc2>`, destroying the memory of the second list is a valid operation, i.e., `Alloc1` giveth and `Alloc2` taketh away.

Second, a list conceptually represented as

A <----> B <----> C <----> D

is actually circular; a link exists between A and D. The list class holds (as its only data member) a private `list::iterator` pointing to *D*, not to *A*! To get to the head of the list, we start at the tail and move forward by one. When this member iterator's `next/previous` pointers refer to itself, the list is empty.

Definition at line 503 of file `stl_list.h`.

5.847.2 Constructor & Destructor Documentation

5.847.2.1 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::list<_Tp, _Alloc>::list () [inline], [noexcept]`

Creates a list with no elements.

Definition at line 585 of file `stl_list.h`.

5.847.2.2 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::list<_Tp, _Alloc>::list (const allocator_type &__a) [inline], [explicit], [noexcept]`

Creates a list with no elements.

Parameters

<code>__a</code>	An allocator object.
------------------	----------------------

Definition at line 596 of file `stl_list.h`.

5.847.2.3 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::list<_Tp, _Alloc>::list (size_type __n, const allocator_type &__a = allocator_type()) [inline], [explicit]`

Creates a list with default constructed elements.

Parameters

<code>__n</code>	The number of elements to initially create.
<code>__a</code>	An allocator object.

This constructor fills the list with `__n` default constructed elements.

Definition at line 609 of file `stl_list.h`.

5.847.2.4 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::list<_Tp, _Alloc>::list (size_type __n, const value_type & __value, const allocator_type & __a = allocator_type()) [inline]`

Creates a list with copies of an exemplar element.

Parameters

<code>__n</code>	The number of elements to initially create.
<code>__value</code>	An element to copy.
<code>__a</code>	An allocator object.

This constructor fills the list with `__n` copies of `__value`.

Definition at line 621 of file `stl_list.h`.

5.847.2.5 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::list<_Tp, _Alloc>::list (const list<_Tp, _Alloc> & __x) [inline]`

List copy constructor.

Parameters

<code>__x</code>	A list of identical element and allocator types.
------------------	--

The newly-created list uses a copy of the allocation object used by `__x`.

Definition at line 648 of file `stl_list.h`.

5.847.2.6 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::list<_Tp, _Alloc>::list (list<_Tp, _Alloc> && __x) [inline], [noexcept]`

List move constructor.

Parameters

<code>__x</code>	A list of identical element and allocator types.
------------------	--

The newly-created list contains the exact contents of `__x`. The contents of `__x` are a valid, but unspecified list.

Definition at line 661 of file `stl_list.h`.

5.847.2.7 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::list<_Tp, _Alloc>::list (initializer_list<value_type> & __l, const allocator_type & __a = allocator_type()) [inline]`

Builds a list from an `initializer_list`.

Parameters

<code>__l</code>	An <code>initializer_list</code> of <code>value_type</code> .
<code>__a</code>	An allocator object.

Create a list consisting of copies of the elements in the `initializer_list __l`. This is linear in `__l.size()`.

Definition at line 672 of file `stl_list.h`.

5.847.2.8 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>> std::list<_Tp, _Alloc>::list (_InputIterator __first, _InputIterator __last, const allocator_type & __a = allocator_type()) [inline]`

Builds a list from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__a</code>	An allocator object.

Create a list consisting of copies of the elements from `[__first, __last)`. This is linear in `N` (where `N` is `distance(__first, __last)`).

Definition at line 705 of file `stl_list.h`.

5.847.3 Member Function Documentation

5.847.3.1 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> template<typename... _Args> _Node* std::list<_Tp, _Alloc>::_M_create_node (_Args &&... __args) [inline], [protected]`

Parameters

<code>__args</code>	An instance of user data.
---------------------	---------------------------

Allocates space for a new node and constructs a copy of `__args` in it.

Definition at line 566 of file `stl_list.h`.

5.847.3.2 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::list<_Tp, _Alloc >::assign (size_type __n, const value_type & __val) [inline]`

Assigns a given value to a list.

Parameters

<code>__n</code>	Number of elements to be assigned.
<code>__val</code>	Value to be assigned.

This function fills a list with `__n` copies of the given value. Note that the assignment completely changes the list and that the resulting list's size is the same as the number of elements assigned. Old data may be lost.

Definition at line 784 of file `stl_list.h`.

5.847.3.3 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> template<typename _InputIterator, typename = std::RequireInputIter<_InputIterator>> void std::list<_Tp, _Alloc >::assign (_InputIterator __first, _InputIterator __last) [inline]`

Assigns a range to a list.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

This function fills a list with copies of the elements in the range `[__first, __last)`.

Note that the assignment completely changes the list and that the resulting list's size is the same as the number of elements assigned. Old data may be lost.

Definition at line 803 of file `stl_list.h`.

5.847.3.4 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::list<_Tp, _Alloc >::assign (initializer_list<value_type> __l) [inline]`

Assigns an `initializer_list` to a list.

Parameters

<code>__l</code>	An <code>initializer_list</code> of <code>value_type</code> .
------------------	---

Replace the contents of the list with copies of the elements in the `initializer_list __l`. This is linear in `__l.size()`.

Definition at line 825 of file `stl_list.h`.

5.847.3.5 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> reference std::list<_Tp, _Alloc>::back ()`
`[inline], [noexcept]`

Returns a read/write reference to the data at the last element of the list.

Definition at line 1025 of file `stl_list.h`.

5.847.3.6 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reference std::list<_Tp, _Alloc>::back ()`
`const [inline], [noexcept]`

Returns a read-only (constant) reference to the data at the last element of the list.

Definition at line 1037 of file `stl_list.h`.

5.847.3.7 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> iterator std::list<_Tp, _Alloc>::begin ()`
`[inline], [noexcept]`

Returns a read/write iterator that points to the first element in the list. Iteration is done in ordinary element order.

Definition at line 840 of file `stl_list.h`.

Referenced by `std::list<_Tp, _Alloc>::insert()`, `std::list<__inp, __rebind_inp>::list()`, `std::list<_Tp, _Alloc>::merge()`, `std::list<_Tp, _Alloc>::operator=()`, `std::operator==()`, `std::list<__inp, __rebind_inp>::reverse()`, `std::list<_Tp, _Alloc>::sort()`, and `std::list<__inp, __rebind_inp>::splice()`.

5.847.3.8 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_iterator std::list<_Tp, _Alloc>::begin () const`
`[inline], [noexcept]`

Returns a read-only (constant) iterator that points to the first element in the list. Iteration is done in ordinary element order.

Definition at line 849 of file `stl_list.h`.

5.847.3.9 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_iterator std::list<_Tp, _Alloc>::cbegin () const`
`[inline], [noexcept]`

Returns a read-only (constant) iterator that points to the first element in the list. Iteration is done in ordinary element order.

Definition at line 913 of file `stl_list.h`.

5.847.3.10 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_iterator std::list<_Tp, _Alloc>::cend () const`
`[inline], [noexcept]`

Returns a read-only (constant) iterator that points one past the last element in the list. Iteration is done in ordinary element order.

Definition at line 922 of file `stl_list.h`.

5.847.3.11 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::list<_Tp, _Alloc>::clear ()`
`[inline], [noexcept]`

Erases all the elements. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1373 of file `stl_list.h`.

5.847.3.12 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reverse_iterator std::list<_Tp, _Alloc>::crbegin () const` `[inline], [noexcept]`

Returns a read-only (constant) reverse iterator that points to the last element in the list. Iteration is done in reverse element order.

Definition at line 931 of file `stl_list.h`.

5.847.3.13 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reverse_iterator std::list<_Tp, _Alloc>::crend () const` `[inline], [noexcept]`

Returns a read-only (constant) reverse iterator that points to one before the first element in the list. Iteration is done in reverse element order.

Definition at line 940 of file `stl_list.h`.

5.847.3.14 `template<typename _Tp, typename _Alloc> template<typename... _Args> list<_Tp, _Alloc>::iterator list::emplace (const_iterator __position, _Args &&... __args)`

Constructs object in list before specified iterator.

Parameters

<code>__position</code>	A <code>const_iterator</code> into the list.
<code>__args</code>	Arguments.

Returns

An iterator that points to the inserted data.

This function will insert an object of type `T` constructed with `T(std::forward<Args>(args)...) before the specified location`. Due to the nature of a list this operation can be done in constant time, and does not invalidate iterators and references.

Definition at line 89 of file `list.tcc`.

5.847.3.15 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> bool std::list<_Tp, _Alloc>::empty () const`
`[inline], [noexcept]`

Returns true if the list is empty. (Thus `begin()` would equal `end()`.)

Definition at line 950 of file `stl_list.h`.

Referenced by `std::list<_Tp, _Alloc>::insert()`, `std::list<__inp, __rebind_inp>::reverse()`, `std::list<_Tp, _Alloc>::sort()`, and `std::list<__inp, __rebind_inp>::splice()`.

5.847.3.16 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> iterator std::list<_Tp, _Alloc>::end ()`
`[inline], [noexcept]`

Returns a read/write iterator that points one past the last element in the list. Iteration is done in ordinary element order.

Definition at line 858 of file `stl_list.h`.

Referenced by `std::list< __inp, __rebind_inp >::list()`, `std::list< _Tp, _Alloc >::merge()`, `std::list< _Tp, _Alloc >::operator=()`, `std::operator==()`, `std::list< __inp, __rebind_inp >::reverse()`, and `std::list< __inp, __rebind_inp >::splice()`.

5.847.3.17 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_iterator std::list<_Tp, _Alloc>::end () const`
`[inline], [noexcept]`

Returns a read-only (constant) iterator that points one past the last element in the list. Iteration is done in ordinary element order.

Definition at line 867 of file `stl_list.h`.

5.847.3.18 `template<typename _Tp, typename _Alloc > list<_Tp, _Alloc>::iterator list::erase (const_iterator __position)`
`[noexcept]`

Remove element at given position.

Parameters

<code>__position</code>	Iterator pointing to element to be erased.
-------------------------	--

Returns

An iterator pointing to the next element (or `end()`).

This function will erase the element at the given position and thus shorten the list by one.

Due to the nature of a list this operation can be done in constant time, and only invalidates iterators/references to the element being removed. The user is also cautioned that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 151 of file `list.tcc`.

References `std::advance()`, `std::begin()`, `std::end()`, and `std::list<_Tp, _Alloc>::resize()`.

5.847.3.19 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> iterator std::list<_Tp, _Alloc>::erase (const_iterator __first, const_iterator __last)`
`[inline], [noexcept]`

Remove a range of elements.

Parameters

<code>__first</code>	Iterator pointing to the first element to be erased.
<code>__last</code>	Iterator pointing to one past the last element to be erased.

Returns

An iterator pointing to the element pointed to by *last* prior to erasing (or end()).

This function will erase the elements in the range [first,last) and shorten the list accordingly.

This operation is linear time in the size of the range and only invalidates iterators/references to the element being removed. The user is also cautioned that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1333 of file `stl_list.h`.

```
5.847.3.20  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> reference std::list< _Tp, _Alloc >::front ( )
           [inline], [noexcept]
```

Returns a read/write reference to the data at the first element of the list.

Definition at line 1009 of file `stl_list.h`.

```
5.847.3.21  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reference std::list< _Tp, _Alloc >::front (
           ) const [inline], [noexcept]
```

Returns a read-only (constant) reference to the data at the first element of the list.

Definition at line 1017 of file `stl_list.h`.

```
5.847.3.22  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> allocator_type std::list< _Tp, _Alloc
           >::get_allocator ( ) const [inline], [noexcept]
```

Get a copy of the memory allocation object.

Definition at line 831 of file `stl_list.h`.

```
5.847.3.23  template<typename _Tp, typename _Alloc > list< _Tp, _Alloc >::iterator list::insert ( const_iterator __position,
           const value_type & __x )
```

Inserts given value into list before specified iterator.

Parameters

<code>__position</code>	A <code>const_iterator</code> into the list.
<code>__x</code>	Data to be inserted.

Returns

An iterator that points to the inserted data.

This function will insert a copy of the given value before the specified location. Due to the nature of a list this operation can be done in constant time, and does not invalidate iterators and references.

Definition at line 102 of file list.tcc.

Referenced by `std::list<_Tp, _Alloc>::insert()`.

5.847.3.24 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> iterator std::list<_Tp, _Alloc>::insert (`
`const_iterator __position, value_type && __x) [inline]`

Inserts given rvalue into list before specified iterator.

Parameters

<code>__position</code>	A <code>const_iterator</code> into the list.
<code>__x</code>	Data to be inserted.

Returns

An iterator that points to the inserted data.

This function will insert a copy of the given rvalue before the specified location. Due to the nature of a list this operation can be done in constant time, and does not invalidate iterators and references.

Definition at line 1185 of file `stl_list.h`.

5.847.3.25 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> iterator std::list<_Tp, _Alloc>::insert (`
`const_iterator __p, initializer_list<value_type> __l) [inline]`

Inserts the contents of an `initializer_list` into list before specified `const_iterator`.

Parameters

<code>__p</code>	A <code>const_iterator</code> into the list.
<code>__l</code>	An <code>initializer_list</code> of <code>value_type</code> .

Returns

An iterator pointing to the first element inserted (or `__position`).

This function will insert copies of the data in the `initializer_list l` into the list before the location specified by `p`.

This operation is linear in the number of elements inserted and does not invalidate iterators and references.

Definition at line 1204 of file `stl_list.h`.

5.847.3.26 `template<typename _Tp, typename _Alloc > list< _Tp, _Alloc >::iterator list::insert (const_iterator __position, size_type __n, const value_type & __x)`

Inserts a number of copies of given data into the list.

Parameters

<code>__position</code>	A const_iterator into the list.
<code>__n</code>	Number of elements to be inserted.
<code>__x</code>	Data to be inserted.

Returns

An iterator pointing to the first element inserted (or `__position`).

This function will insert a specified number of copies of the given data before the location specified by *position*.

This operation is linear in the number of elements inserted and does not invalidate iterators and references.

Definition at line 117 of file list.tcc.

References `std::list< _Tp, _Alloc >::begin()`, and `std::list< _Tp, _Alloc >::insert()`.

5.847.3.27 `template<typename _Tp, typename _Alloc > template<typename _InputIterator, typename > list< _Tp, _Alloc >::iterator list::insert (const_iterator __position, _InputIterator __first, _InputIterator __last)`

Inserts a range into the list.

Parameters

<code>__position</code>	A const_iterator into the list.
<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Returns

An iterator pointing to the first element inserted (or `__position`).

This function will insert copies of the data in the range *[first,last)* into the list before the location specified by *position*.

This operation is linear in the number of elements inserted and does not invalidate iterators and references.

Definition at line 133 of file list.tcc.

References `std::list< _Tp, _Alloc >::begin()`, and `std::list< _Tp, _Alloc >::empty()`.

5.847.3.28 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> size_type std::list<_Tp, _Alloc>::max_size ()`
`const [inline],[noexcept]`

Returns the size() of the largest possible list.

Definition at line 960 of file `stl_list.h`.

5.847.3.29 `template<typename _Tp, typename _Alloc > void list::merge (list<_Tp, _Alloc> && __x)`

Merge sorted lists.

Parameters

<code>__x</code>	Sorted list to merge.
------------------	-----------------------

Assumes that both `__x` and this list are sorted according to operator<(). Merges elements of `__x` into this list in sorted order, leaving `__x` empty when complete. Elements in this list precede elements in `__x` that are equal.

Definition at line 374 of file `list.tcc`.

References `std::__addressof()`, `std::begin()`, `std::list<_Tp, _Alloc>::begin()`, `std::end()`, and `std::list<_Tp, _Alloc>::end()`.

Referenced by `std::list<_Tp, _Alloc>::sort()`.

5.847.3.30 `template<typename _Tp, typename _Alloc > template<typename _StrictWeakOrdering > void list::merge (list<_Tp, _Alloc> && __x, _StrictWeakOrdering __comp)`

Merge sorted lists according to comparison function.

Template Parameters

<code>_StrictWeakOrdering</code>	Comparison function defining sort order.
----------------------------------	--

Parameters

<code>__x</code>	Sorted list to merge.
<code>__comp</code>	Comparison functor.

Assumes that both `__x` and this list are sorted according to `StrictWeakOrdering`. Merges elements of `__x` into this list in sorted order, leaving `__x` empty when complete. Elements in this list precede elements in `__x` that are equivalent according to `StrictWeakOrdering()`.

Definition at line 411 of file `list.tcc`.

References `std::__addressof()`, `std::begin()`, `std::list<_Tp, _Alloc>::begin()`, `std::end()`, `std::list<_Tp, _Alloc>::end()`, and `std::list<_Tp, _Alloc>::sort()`.

5.847.3.31 `template<typename _Tp, typename _Alloc> list< _Tp, _Alloc> & list::operator= (const list< _Tp, _Alloc> & __x)`

List assignment operator.

No explicit dtor needed as the _Base dtor takes care of things. The _Base dtor only erases the elements, and note that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Parameters

<code>__x</code>	A list of identical element and allocator types.
------------------	--

All the elements of `__x` are copied, but unlike the copy constructor, the allocator object is not copied.

Definition at line 267 of file list.tcc.

References `std::__addressof()`, `std::begin()`, `std::list< _Tp, _Alloc>::begin()`, `std::end()`, `std::list< _Tp, _Alloc>::end()`, and `std::list< _Tp, _Alloc>::remove()`.

Referenced by `std::list< _Tp, _Alloc>::resize()`.

5.847.3.32 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> list& std::list< _Tp, _Alloc>::operator= (list< _Tp, _Alloc> && __x) [inline], [noexcept]`

List move assignment operator.

Parameters

<code>__x</code>	A list of identical element and allocator types.
------------------	--

The contents of `__x` are moved into this list (without copying). `__x` is a valid, but unspecified list

Definition at line 748 of file stl_list.h.

5.847.3.33 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> list& std::list< _Tp, _Alloc>::operator= (initializer_list< value_type> __l) [inline]`

List initializer list assignment operator.

Parameters

<code>__l</code>	An initializer_list of value_type.
------------------	------------------------------------

Replace the contents of the list with copies of the elements in the initializer_list __l/. This is linear in l.size().

Definition at line 766 of file stl_list.h.

5.847.3.34 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::list<_Tp, _Alloc>::pop_back ()`
`[inline], [noexcept]`

Removes last element.

This is a typical stack operation. It shrinks the list by one. Due to the nature of a list this operation can be done in constant time, and only invalidates iterators/references to the element being removed.

Note that no data is returned, and if the last element's data is needed, it should be retrieved before pop_back() is called.

Definition at line 1123 of file stl_list.h.

5.847.3.35 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::list<_Tp, _Alloc>::pop_front ()`
`[inline], [noexcept]`

Removes first element.

This is a typical stack operation. It shrinks the list by one. Due to the nature of a list this operation can be done in constant time, and only invalidates iterators/references to the element being removed.

Note that no data is returned, and if the first element's data is needed, it should be retrieved before pop_front() is called.

Definition at line 1083 of file stl_list.h.

5.847.3.36 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::list<_Tp, _Alloc>::push_back (const`
`value_type & __x) [inline]`

Add data to the end of the list.

Parameters

<code>__x</code>	Data to be added.
------------------	-------------------

This is a typical stack operation. The function creates an element at the end of the list and assigns the given data to it. Due to the nature of a list this operation can be done in constant time, and does not invalidate iterators and references.

Definition at line 1097 of file stl_list.h.

5.847.3.37 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::list<_Tp, _Alloc>::push_front (const`
`value_type & __x) [inline]`

Add data to the front of the list.

Parameters

<code>_↔</code>	Data to be added.
<code>_X</code>	

This is a typical stack operation. The function creates an element at the front of the list and assigns the given data to it. Due to the nature of a list this operation can be done in constant time, and does not invalidate iterators and references.

Definition at line 1056 of file `stl_list.h`.

5.847.3.38 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> reverse_iterator std::list<_Tp, _Alloc>::rbegin() [inline], [noexcept]`

Returns a read/write reverse iterator that points to the last element in the list. Iteration is done in reverse element order.

Definition at line 876 of file `stl_list.h`.

5.847.3.39 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reverse_iterator std::list<_Tp, _Alloc>::rbegin() const [inline], [noexcept]`

Returns a read-only (constant) reverse iterator that points to the last element in the list. Iteration is done in reverse element order.

Definition at line 885 of file `stl_list.h`.

5.847.3.40 `template<typename _Tp, typename _Alloc > void list::remove(const _Tp & __value)`

Remove all elements equal to `value`.

Parameters

<code>__value</code>	The value to remove.
----------------------	----------------------

Removes every element in the list equal to `value`. Remaining elements stay in list order. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 325 of file `list.tcc`.

References `std::__addressof()`, `std::begin()`, `std::end()`, and `std::list<_Tp, _Alloc>::unique()`.

Referenced by `std::list<_Tp, _Alloc>::operator=()`.

5.847.3.41 `template<typename _Tp, typename _Alloc > template<typename _Predicate > void list::remove_if(_Predicate __pred)`

Remove all elements satisfying a predicate.

Template Parameters

<code>_Predicate</code>	Unary predicate function or object.
-------------------------	-------------------------------------

Removes every element in the list for which the predicate returns true. Remaining elements stay in list order. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 484 of file list.tcc.

References `std::begin()`, `std::end()`, and `std::list<_Tp, _Alloc>::unique()`.

Referenced by `std::list<_Tp, _Alloc>::sort()`.

5.847.3.42 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> reverse_iterator std::list<_Tp, _Alloc>::rend () [inline], [noexcept]`

Returns a read/write reverse iterator that points to one before the first element in the list. Iteration is done in reverse element order.

Definition at line 894 of file stl_list.h.

5.847.3.43 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reverse_iterator std::list<_Tp, _Alloc>::rend () const [inline], [noexcept]`

Returns a read-only (constant) reverse iterator that points to one before the first element in the list. Iteration is done in reverse element order.

Definition at line 903 of file stl_list.h.

5.847.3.44 `template<typename _Tp, typename _Alloc> void list::resize (size_type __new_size)`

Resizes the list to the specified number of elements.

Parameters

<code>__new_size</code>	Number of elements the list should contain.
-------------------------	---

This function will resize the list to the specified number of elements. If the number is smaller than the list's current size the list is truncated, otherwise default constructed elements are appended.

Definition at line 230 of file list.tcc.

References `std::end()`.

Referenced by `std::list<_Tp, _Alloc>::erase()`, and `std::list<_Tp, _Alloc>::resize()`.

5.847.3.45 `template<typename _Tp, typename _Alloc> void list::resize (size_type __new_size, const value_type & __x)`

Resizes the list to the specified number of elements.

Parameters

<code>__new_size</code>	Number of elements the list should contain.
<code>__x</code>	Data with which new elements should be populated.

This function will resize the list to the specified number of elements. If the number is smaller than the list's current size the list is truncated, otherwise the list is extended and new elements are populated with given data.

Definition at line 242 of file list.tcc.

References `std::end()`, `std::list< _Tp, _Alloc >::operator=()`, and `std::list< _Tp, _Alloc >::resize()`.

5.847.3.46 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::list< _Tp, _Alloc >::reverse ()`
`[inline], [noexcept]`

Reverse the elements in list.

Reverse the order of elements in the list in linear time.

Definition at line 1657 of file stl_list.h.

5.847.3.47 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> size_type std::list< _Tp, _Alloc >::size () const`
`[inline], [noexcept]`

Returns the number of elements in the list.

Definition at line 955 of file stl_list.h.

Referenced by `std::operator==()`.

5.847.3.48 `template<typename _Tp, typename _Alloc > void list::sort ()`

Sort the elements.

Sorts the elements of this list in NlogN time. Equivalent elements remain in list order.

Definition at line 446 of file list.tcc.

References `std::begin()`, `std::list< _Tp, _Alloc >::begin()`, `std::list< _Tp, _Alloc >::empty()`, `std::list< _Tp, _Alloc >::merge()`, `std::list< _Tp, _Alloc >::remove_if()`, `std::list< _Tp, _Alloc >::splice()`, and `std::list< _Tp, _Alloc >::swap()`.

Referenced by `std::list< _Tp, _Alloc >::merge()`, and `std::list< _Tp, _Alloc >::unique()`.

5.847.3.49 `template<typename _Tp, typename _Alloc > template<typename _StrictWeakOrdering > void list::sort (`
`_StrictWeakOrdering __comp)`

Sort the elements according to comparison function.

Sorts the elements of this list in NlogN time. Equivalent elements remain in list order.

Definition at line 523 of file list.tcc.

References `std::begin()`, `std::list< _Tp, _Alloc >::begin()`, `std::list< _Tp, _Alloc >::empty()`, `std::list< _Tp, _Alloc >::merge()`, `std::list< _Tp, _Alloc >::splice()`, and `std::list< _Tp, _Alloc >::swap()`.

5.847.3.50 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::list< _Tp, _Alloc >::splice (`
`const_iterator __position, list< _Tp, _Alloc > && __x) [inline], [noexcept]`

Insert contents of another list.

Parameters

<code>__position</code>	Iterator referencing the element to insert before.
<code>__x</code>	Source list.

The elements of `__x` are inserted in constant time in front of the element referenced by `__position`. `__x` becomes an empty list.

Requires this `!= __x`.

Definition at line 1393 of file `stl_list.h`.

Referenced by `std::list<_Tp, _Alloc>::sort()`.

```
5.847.3.51  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::list<_Tp, _Alloc>::splice (
            const_iterator __position, list<_Tp, _Alloc> && __x, const_iterator __i )  [inline], [noexcept]
```

Insert element from another list.

Parameters

<code>__position</code>	Const_iterator referencing the element to insert before.
<code>__x</code>	Source list.
<code>__i</code>	Const_iterator referencing the element to move.

Removes the element in list `__x` referenced by `__i` and inserts it into the current list before `__position`.

Definition at line 1428 of file `stl_list.h`.

```
5.847.3.52  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::list<_Tp, _Alloc>::splice (
            const_iterator __position, list<_Tp, _Alloc> & __x, const_iterator __i )  [inline], [noexcept]
```

Insert element from another list.

Parameters

<code>__position</code>	Const_iterator referencing the element to insert before.
<code>__x</code>	Source list.
<code>__i</code>	Const_iterator referencing the element to move.

Removes the element in list `__x` referenced by `__i` and inserts it into the current list before `__position`.

Definition at line 1470 of file `stl_list.h`.

```
5.847.3.53  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::list<_Tp, _Alloc>::splice (
            const_iterator __position, list<_Tp, _Alloc> && __x, const_iterator __first, const_iterator __last )
            [inline], [noexcept]
```

Insert range from another list.

Parameters

<code>__position</code>	Const_iterator referencing the element to insert before.
<code>__x</code>	Source list.
<code>__first</code>	Const_iterator referencing the start of range in x.
<code>__last</code>	Const_iterator referencing the end of range in x.

Removes elements in the range [`__first`,`__last`) and inserts them before `__position` in constant time.

Undefined if `__position` is in [`__first`,`__last`).

Definition at line 1489 of file `stl_list.h`.

```
5.847.3.54 template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::list< _Tp, _Alloc >::splice (
    const_iterator __position, list< _Tp, _Alloc > & __x, const_iterator __first, const_iterator __last )
    [inline], [noexcept]
```

Insert range from another list.

Parameters

<code>__position</code>	Const_iterator referencing the element to insert before.
<code>__x</code>	Source list.
<code>__first</code>	Const_iterator referencing the start of range in x.
<code>__last</code>	Const_iterator referencing the end of range in x.

Removes elements in the range [`__first`,`__last`) and inserts them before `__position` in constant time.

Undefined if `__position` is in [`__first`,`__last`).

Definition at line 1539 of file `stl_list.h`.

```
5.847.3.55 template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::list< _Tp, _Alloc >::swap ( list< _Tp,
    _Alloc > & __x ) [inline], [noexcept]
```

Swaps data with another list.

Parameters

<code>__x</code>	A list of the same element and allocator types.
------------------	---

This exchanges the elements between two lists in constant time. Note that the global `std::swap()` function is specialized such that `std::swap(l1,l2)` will feed to this function.

Definition at line 1353 of file `stl_list.h`.

Referenced by `std::list< _Tp, _Alloc >::sort()`.

5.847.3.56 `template<typename _Tp, typename _Alloc > void list::unique ()`

Remove consecutive duplicate elements.

For each consecutive set of elements with the same value, remove all but the first one. Remaining elements stay in list order. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 353 of file list.tcc.

References `std::begin()`, and `std::end()`.

Referenced by `std::list< _Tp, _Alloc >::remove()`, and `std::list< _Tp, _Alloc >::remove_if()`.

5.847.3.57 `template<typename _Tp, typename _Alloc > template<typename _BinaryPredicate > void list::unique (_BinaryPredicate __binary_pred)`

Remove consecutive elements satisfying a predicate.

Template Parameters

<code>_BinaryPredicate</code>	Binary predicate function or object.
-------------------------------	--------------------------------------

For each consecutive set of elements `[first,last)` that satisfy `predicate(first,i)` where `i` is an iterator in `[first,last)`, remove all but the first one. Remaining elements stay in list order. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 502 of file list.tcc.

References `std::begin()`, `std::end()`, and `std::list< _Tp, _Alloc >::sort()`.

The documentation for this class was generated from the following files:

- [stl_list.h](#)
- [list.tcc](#)

5.848 `std::locale` Class Reference

Classes

- class [facet](#)
- class [id](#)

Public Types

- typedef int [category](#)

Public Member Functions

- [locale](#) () throw ()
- [locale](#) (const [locale](#) &__other) throw ()
- [locale](#) (const char *__s)
- [locale](#) (const [locale](#) &__base, const char *__s, [category](#) __cat)
- [locale](#) (const [std::string](#) &__s)
- [locale](#) (const [locale](#) &__base, const [std::string](#) &__s, [category](#) __cat)
- [locale](#) (const [locale](#) &__base, const [locale](#) &__add, [category](#) __cat)
- template<typename _Facet >
[locale](#) (const [locale](#) &__other, _Facet *__f)
- [~locale](#) () throw ()
- template<typename _Facet >
[locale combine](#) (const [locale](#) &__other) const
- [_GLIBCXX_DEFAULT_ABI_TAG string name](#) () const
- bool [operator!=](#) (const [locale](#) &__other) const throw ()
- template<typename _CharT, typename _Traits, typename _Alloc >
bool [operator\(\)](#) (const [basic_string](#)< _CharT, _Traits, _Alloc > &__s1, const [basic_string](#)< _CharT, _Traits, _Alloc > &__s2) const
- template<typename _Char, typename _Traits, typename _Alloc >
bool [operator\(\)](#) (const [basic_string](#)< _Char, _Traits, _Alloc > &__s1, const [basic_string](#)< _Char, _Traits, _Alloc > &__s2) const
- const [locale](#) & [operator=](#) (const [locale](#) &__other) throw ()
- bool [operator==](#) (const [locale](#) &__other) const throw ()

Static Public Member Functions

- static const [locale](#) & [classic](#) ()
- static [locale global](#) (const [locale](#) &__loc)

Static Public Attributes

- static const [category none](#)
- static const [category ctype](#)
- static const [category numeric](#)
- static const [category collate](#)
- static const [category time](#)
- static const [category monetary](#)
- static const [category messages](#)
- static const [category all](#)

Friends

- template<typename _Cache >
struct [__use_cache](#)
- class [_Impl](#)
- class [facet](#)
- template<typename _Facet >
bool [has_facet](#) (const [locale](#) &) throw ()
- template<typename _Facet >
const _Facet & [use_facet](#) (const [locale](#) &)

5.848.1 Detailed Description

Container class for localization functionality.

The locale class is first a class wrapper for C library locales. It is also an extensible container for user-defined localization. A locale is a collection of facets that implement various localization features such as money, time, and number printing.

Constructing C++ locales does not change the C library locale.

This library supports efficient construction and copying of locales through a reference counting implementation of the locale class.

Definition at line 62 of file locale_classes.h.

5.848.2 Member Typedef Documentation

5.848.2.1 `typedef int std::locale::category`

Definition of locale::category.

Definition at line 67 of file locale_classes.h.

5.848.3 Constructor & Destructor Documentation

5.848.3.1 `std::locale::locale () throw`

Default constructor.

Constructs a copy of the global locale. If no locale has been explicitly set, this is the C locale.

Referenced by combine(), locale(), and operator!=().

5.848.3.2 `std::locale::locale (const locale & __other) throw`

Copy constructor.

Constructs a copy of *other*.

Parameters

<code>__other</code>	The locale to copy.
----------------------	---------------------

5.848.3.3 `std::locale::locale (const char * __s) [explicit]`

Named locale constructor.

Constructs a copy of the named C library locale.

Parameters

<code>__s</code>	Name of the locale to construct.
------------------	----------------------------------

Exceptions

<code>std::runtime_error</code>	if <code>__s</code> is null or an undefined locale.
---------------------------------	---

5.848.3.4 `std::locale::locale (const locale & __base, const char * __s, category __cat)`

Construct locale with facets from another locale.

Constructs a copy of the locale *base*. The facets specified by *cat* are replaced with those from the locale named by *s*. If *base* is named, this locale instance will also be named.

Parameters

<code>__base</code>	The locale to copy.
<code>__s</code>	Name of the locale to use facets from.
<code>__cat</code>	Set of categories defining the facets to use from <code>__s</code> .

Exceptions

<code>std::runtime_error</code>	if <code>__s</code> is null or an undefined locale.
---------------------------------	---

5.848.3.5 `std::locale::locale (const std::string & __s) [inline],[explicit]`

Named locale constructor.

Constructs a copy of the named C library locale.

Parameters

<code>__s</code>	Name of the locale to construct.
------------------	----------------------------------

Exceptions

<code>std::runtime_error</code>	if <code>__s</code> is an undefined locale.
---------------------------------	---

Definition at line 163 of file `locale_classes.h`.

5.848.3.6 `std::locale::locale (const locale & __base, const std::string & __s, category __cat)` `[inline]`

Construct locale with facets from another locale.

Constructs a copy of the locale *base*. The facets specified by *cat* are replaced with those from the locale named by *s*. If *base* is named, this locale instance will also be named.

Parameters

<code>__base</code>	The locale to copy.
<code>__s</code>	Name of the locale to use facets from.
<code>__cat</code>	Set of categories defining the facets to use from <code>__s</code> .

Exceptions

<code>std::runtime_error</code>	if <code>__s</code> is an undefined locale.
---------------------------------	---

Definition at line 177 of file `locale_classes.h`.

References `combine()`, `locale()`, `name()`, `operator=()`, `operator==()`, and `~locale()`.

5.848.3.7 `std::locale::locale (const locale & __base, const locale & __add, category __cat)`

Construct locale with facets from another locale.

Constructs a copy of the locale *base*. The facets specified by *cat* are replaced with those from the locale *add*. If *base* and *add* are named, this locale instance will also be named.

Parameters

<code>__base</code>	The locale to copy.
<code>__add</code>	The locale to use facets from.
<code>__cat</code>	Set of categories defining the facets to use from <i>add</i> .

5.848.3.8 `template<typename _Facet > std::locale::locale (const locale & __other, _Facet * __f)`

Construct locale with another facet.

Constructs a copy of the locale `__other`. The facet `__f` is added to `__other`, replacing an existing facet of type `Facet` if there is one. If `__f` is null, this locale is a copy of `__other`.

Parameters

<code>__other</code>	The locale to copy.
<code>__f</code>	The facet to add in.

Definition at line 45 of file locale_classes.tcc.

References combine().

5.848.3.9 std::locale::~~locale () throw

Locale destructor.

Referenced by locale().

5.848.4 Member Function Documentation

5.848.4.1 static const locale& std::locale::classic () [static]

Return reference to the C locale.

Referenced by operator!==().

5.848.4.2 template<typename _Facet > locale std::locale::combine (const locale & __other) const

Construct locale with another facet.

Constructs and returns a new copy of this locale. Adds or replaces an existing facet of type Facet from the locale *other* into the new locale.

Template Parameters

<code>_Facet</code>	The facet type to copy from other
---------------------	-----------------------------------

Parameters

<code>__other</code>	The locale to copy from.
----------------------	--------------------------

Returns

Newly constructed locale.

Exceptions

<code>std::runtime_error</code>	if <code>__other</code> has no facet of type <code>_Facet</code> .
---------------------------------	--

Definition at line 63 of file locale_classes.tcc.

References `std::basic_string< _CharT, _Traits, _Alloc >::data()`, `std::basic_string< _CharT, _Traits, _Alloc >::length()`, `locale()`, and `operator()()`.

Referenced by locale().

5.848.4.3 static locale std::locale::global (const locale & __loc) [static]

Set global locale.

This function sets the global locale to the argument and returns a copy of the previous global locale. If the argument has a name, it will also call std::setlocale(LC_ALL, loc.name()).

Parameters

<code>__loc</code>	The new locale to make global.
--------------------	--------------------------------

Returns

Copy of the old global locale.

Referenced by operator!=().

5.848.4.4 _GLIBCXX_DEFAULT_ABI_TAG string std::locale::name () const

Return locale name.

Returns

Locale name or "*" if unnamed.

Referenced by locale().

5.848.4.5 bool std::locale::operator!= (const locale & __other) const throw () [inline]

Locale inequality.

Parameters

<code>__other</code>	The locale to compare against.
----------------------	--------------------------------

Returns

! (*this == __other)

Definition at line 264 of file locale_classes.h.

References classic(), global(), locale(), operator()(), and operator==().

5.848.4.6 template<typename _Char, typename _Traits, typename _Alloc> bool std::locale::operator() (const basic_string<_Char, _Traits, _Alloc> & __s1, const basic_string<_Char, _Traits, _Alloc> & __s2) const

Compare two strings according to collate.

Template operator to compare two strings using the compare function of the collate facet in this locale. One use is to provide the locale to the sort function. For example, a vector `v` of strings could be sorted according to locale `loc` by doing:

```
std::sort(v.begin(), v.end(), loc);
```

Parameters

<code>__s1</code>	First string to compare.
<code>__s2</code>	Second string to compare.

Returns

True if `collate<_Char> facet` compares `__s1 < __s2`, else false.

Referenced by `combine()`, and `operator!=()`.

5.848.4.7 `const locale& std::locale::operator= (const locale & __other) throw (`

Assignment operator.

Set this locale to be a copy of *other*.

Parameters

<code>__other</code>	The locale to copy.
----------------------	---------------------

Returns

A reference to this locale.

Referenced by `std::locale::facet::facet()`, `std::locale::id::id()`, and `locale()`.

5.848.4.8 `bool std::locale::operator== (const locale & __other) const throw (`

Locale equality.

Parameters

<code>__other</code>	The locale to compare against.
----------------------	--------------------------------

Returns

True if *other* and this refer to the same locale instance, are copies, or have the same name. False otherwise.

Referenced by `locale()`, and `operator!=()`.

5.848.5 Friends And Related Function Documentation

5.848.5.1 `template<typename _Facet > bool has_facet (const locale &) throw` `[friend]`

Test for the presence of a facet.

`has_facet` tests the locale argument for the presence of the facet type provided as the template parameter. Facets derived from the facet parameter will also return true.

Template Parameters

<code>_Facet</code>	The facet type to test the presence of.
---------------------	---

Parameters

<code>__loc</code>	The locale to test.
--------------------	---------------------

Returns

true if `__loc` contains a facet of type `_Facet`, else false.

Definition at line 104 of file `locale_classes.tcc`.

Referenced by `std::locale::id::id()`.

5.848.5.2 `template<typename _Facet > const _Facet& use_facet (const locale &)` `[friend]`

Return a facet.

`use_facet` looks for and returns a reference to a facet of type `Facet` where `Facet` is the template parameter. If `has_facet(locale)` is true, there is a suitable facet to return. It throws `std::bad_cast` if the locale doesn't contain a facet of type `Facet`.

Template Parameters

<code>_Facet</code>	The facet type to access.
---------------------	---------------------------

Parameters

<code>__loc</code>	The locale to use.
--------------------	--------------------

Returns

Reference to facet of type `Facet`.

Exceptions

<code>std::bad_cast</code>	if <code>__loc</code> doesn't contain a facet of type <code>_Facet</code> .
----------------------------	---

Definition at line 132 of file `locale_classes.tcc`.

Referenced by `std::locale::id::id()`.

5.848.6 Member Data Documentation

5.848.6.1 **const category** `std::locale::all` `[static]`

Category values.

The standard category values are none, ctype, numeric, collate, time, monetary, and messages. They form a bitmask that supports union and intersection. The category all is the union of these values.

NB: Order must match `_S_facet_categories` definition in `locale.cc`

Definition at line 105 of file `locale_classes.h`.

5.848.6.2 **const category** `std::locale::collate` `[static]`

Category values.

The standard category values are none, ctype, numeric, collate, time, monetary, and messages. They form a bitmask that supports union and intersection. The category all is the union of these values.

NB: Order must match `_S_facet_categories` definition in `locale.cc`

Definition at line 101 of file `locale_classes.h`.

5.848.6.3 **const category** `std::locale::ctype` `[static]`

Category values.

The standard category values are none, ctype, numeric, collate, time, monetary, and messages. They form a bitmask that supports union and intersection. The category all is the union of these values.

NB: Order must match `_S_facet_categories` definition in `locale.cc`

Definition at line 99 of file `locale_classes.h`.

5.848.6.4 **const category** `std::locale::messages` `[static]`

Category values.

The standard category values are none, ctype, numeric, collate, time, monetary, and messages. They form a bitmask that supports union and intersection. The category all is the union of these values.

NB: Order must match `_S_facet_categories` definition in `locale.cc`

Definition at line 104 of file `locale_classes.h`.

5.848.6.5 `const category std::locale::monetary` `[static]`

Category values.

The standard category values are none, ctype, numeric, collate, time, monetary, and messages. They form a bitmask that supports union and intersection. The category all is the union of these values.

NB: Order must match `_S_facet_categories` definition in `locale.cc`

Definition at line 103 of file `locale_classes.h`.

5.848.6.6 `const category std::locale::none` `[static]`

Category values.

The standard category values are none, ctype, numeric, collate, time, monetary, and messages. They form a bitmask that supports union and intersection. The category all is the union of these values.

NB: Order must match `_S_facet_categories` definition in `locale.cc`

Definition at line 98 of file `locale_classes.h`.

5.848.6.7 `const category std::locale::numeric` `[static]`

Category values.

The standard category values are none, ctype, numeric, collate, time, monetary, and messages. They form a bitmask that supports union and intersection. The category all is the union of these values.

NB: Order must match `_S_facet_categories` definition in `locale.cc`

Definition at line 100 of file `locale_classes.h`.

5.848.6.8 `const category std::locale::time` `[static]`

Category values.

The standard category values are none, ctype, numeric, collate, time, monetary, and messages. They form a bitmask that supports union and intersection. The category all is the union of these values.

NB: Order must match `_S_facet_categories` definition in `locale.cc`

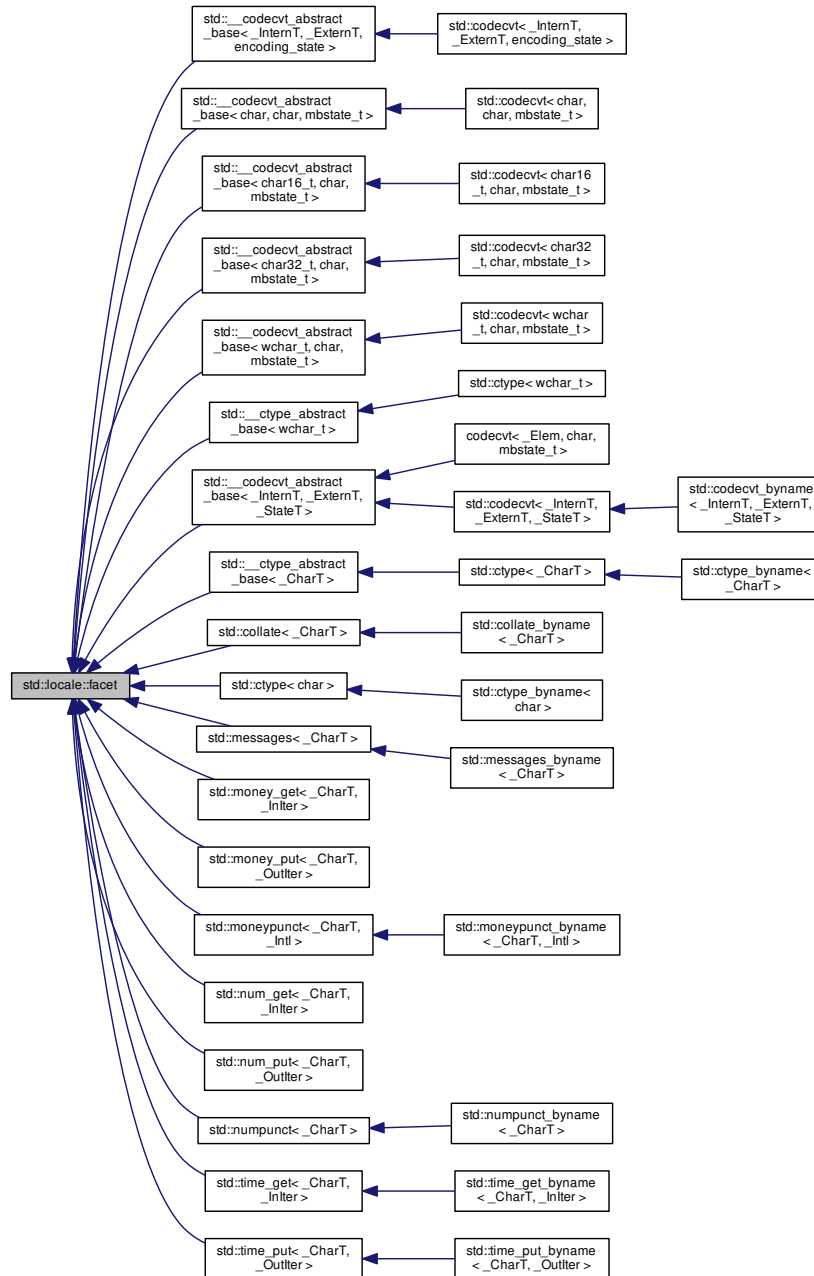
Definition at line 102 of file `locale_classes.h`.

The documentation for this class was generated from the following files:

- [locale_classes.h](#)
- [locale_classes.tcc](#)

5.849 std::locale::facet Class Reference

Inheritance diagram for std::locale::facet:



Protected Member Functions

- [facet](#) (size_t __refs=0) throw ()

- **facet** (const facet &)=delete
- virtual ~facet ()
- **facet** & **operator=** (const facet &)=delete

Static Protected Member Functions

- static __c_locale **_S_clone_c_locale** (__c_locale &__cloc) throw ()
- static void **_S_create_c_locale** (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale &__cloc)
- static __c_locale **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static __c_locale **_S_lc_ctype_c_locale** (__c_locale __cloc, const char *__s)

Friends

- class **locale**
- class **locale::_Impl**

5.849.1 Detailed Description

Localization functionality base class.

The facet class is the base class for a localization feature, such as money, time, and number printing. It provides common support for facets and reference management.

Facets may not be copied or assigned.

Definition at line 371 of file locale_classes.h.

5.849.2 Constructor & Destructor Documentation

5.849.2.1 `std::locale::facet::facet (size_t __refs = 0) throw ()` `[inline], [explicit], [protected]`

Facet constructor.

This is the constructor provided by the standard. If refs is 0, the facet is destroyed when the last referencing locale is destroyed. Otherwise the facet will never be destroyed.

Parameters

<code>__refs</code>	The initial value for reference count.
---------------------	--

Definition at line 403 of file locale_classes.h.

References `std::locale::operator=()`.

Referenced by `std::__ctype_abstract_base< wchar_t >::narrow()`.

5.849.2.2 `virtual std::locale::facet::~~facet () [protected],[virtual]`

Facet destructor.

The documentation for this class was generated from the following file:

- [locale_classes.h](#)

5.850 `std::locale::id` Class Reference

Public Member Functions

- `id ()`
- `size_t _M_id () const throw ()`

Friends

- `template<typename _Facet >`
`bool has_facet (const locale &) throw ()`
- `class locale`
- `class locale::_Impl`
- `template<typename _Facet >`
`const _Facet & use_facet (const locale &)`

5.850.1 Detailed Description

Facet ID class.

The ID class provides facets with an index used to identify them. Every facet class must define a public static member `locale::id`, or be derived from a facet that provides this member, otherwise the facet cannot be used in a locale. The `locale::id` ensures that each class type gets a unique identifier.

Definition at line 482 of file `locale_classes.h`.

5.850.2 Constructor & Destructor Documentation

5.850.2.1 `std::locale::id::id () [inline]`

Constructor.

Definition at line 513 of file `locale_classes.h`.

References `std::locale::has_facet`, `std::locale::operator=()`, and `std::locale::use_facet`.

5.850.3 Friends And Related Function Documentation

5.850.3.1 `template<typename _Facet > bool has_facet (const locale &) throw) [friend]`

Test for the presence of a facet.

`has_facet` tests the locale argument for the presence of the facet type provided as the template parameter. Facets derived from the facet parameter will also return true.

Template Parameters

<code>_Facet</code>	The facet type to test the presence of.
---------------------	---

Parameters

<code>__loc</code>	The locale to test.
--------------------	---------------------

Returns

true if `__loc` contains a facet of type `_Facet`, else false.

Definition at line 104 of file `locale_classes.tcc`.

5.850.3.2 `template<typename _Facet> const _Facet& use_facet (const locale &) [friend]`

Return a facet.

`use_facet` looks for and returns a reference to a facet of type `Facet` where `Facet` is the template parameter. If `has_facet(locale)` is true, there is a suitable facet to return. It throws `std::bad_cast` if the locale doesn't contain a facet of type `Facet`.

Template Parameters

<code>_Facet</code>	The facet type to access.
---------------------	---------------------------

Parameters

<code>__loc</code>	The locale to use.
--------------------	--------------------

Returns

Reference to facet of type `Facet`.

Exceptions

<code>std::bad_cast</code>	if <code>__loc</code> doesn't contain a facet of type <code>_Facet</code> .
----------------------------	---

Definition at line 132 of file `locale_classes.tcc`.

The documentation for this class was generated from the following file:

- [locale_classes.h](#)

5.851 `std::lock_guard<_Mutex>` Class Template Reference

Public Types

- typedef `_Mutex` **`mutex_type`**

Public Member Functions

- **`lock_guard`** (`mutex_type` &__m)
- **`lock_guard`** (`mutex_type` &__m, [adopt_lock_t](#))
- **`lock_guard`** (const [lock_guard](#) &)=delete
- **`lock_guard`** & **`operator=`** (const [lock_guard](#) &)=delete

5.851.1 Detailed Description

```
template<typename _Mutex>
class std::lock_guard<_Mutex>
```

A simple scoped lock type.

A `lock_guard` controls mutex ownership within a scope, releasing ownership in the destructor.

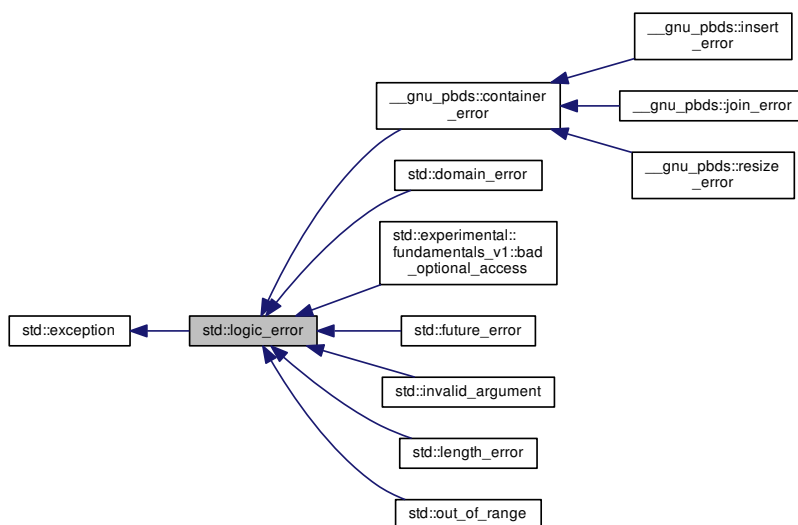
Definition at line 156 of file `std_mutex.h`.

The documentation for this class was generated from the following file:

- [std_mutex.h](#)

5.852 `std::logic_error` Class Reference

Inheritance diagram for `std::logic_error`:



Public Member Functions

- [logic_error](#) (const [string](#) & __arg) _GLIBCXX_TXN_SAFE
- **logic_error** (const char *) _GLIBCXX_TXN_SAFE
- virtual const char * [what](#) () const _GLIBCXX_TXN_SAFE_DYN noexcept

5.852.1 Detailed Description

One of two subclasses of exception.

Logic errors represent problems in the internal logic of a program; in theory, these are preventable, and even detectable before the program runs (e.g., violations of class invariants).

Definition at line 113 of file `stdexcept`.

5.852.2 Constructor & Destructor Documentation

5.852.2.1 `std::logic_error::logic_error (const string & __arg) [explicit]`

Takes a character string describing the error.

5.852.3 Member Function Documentation

5.852.3.1 `virtual const char* std::logic_error::what () const [virtual], [noexcept]`

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

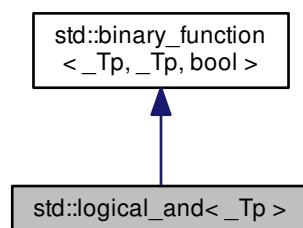
Reimplemented in [std::future_error](#).

The documentation for this class was generated from the following file:

- [stdexcept](#)

5.853 `std::logical_and< _Tp >` Struct Template Reference

Inheritance diagram for `std::logical_and< _Tp >`:



Public Types

- typedef `_Tp` [first_argument_type](#)
- typedef `bool` [result_type](#)
- typedef `_Tp` [second_argument_type](#)

Public Member Functions

- `_GLIBCXX14_CONSTEXPR` `bool` **operator()** (`const _Tp &__x`, `const _Tp &__y`) `const`

5.853.1 Detailed Description

```
template<typename _Tp>
struct std::logical_and<_Tp >
```

One of the [Boolean operations functors](#).

Definition at line 513 of file `stl_function.h`.

5.853.2 Member Typedef Documentation

5.853.2.1 `typedef _Tp std::binary_function<_Tp, _Tp, bool>::first_argument_type` `[inherited]`

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.853.2.2 `typedef bool std::binary_function<_Tp, _Tp, bool>::result_type` `[inherited]`

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.853.2.3 `typedef _Tp std::binary_function<_Tp, _Tp, bool>::second_argument_type` `[inherited]`

`second_argument_type` is the type of the second argument

Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.854 `std::logical_and< void >` Struct Template Reference

Public Types

- typedef `__is_transparent` **is_transparent**

Public Member Functions

- template<typename `_Tp`, typename `_Up`>
`_GLIBCXX14_CONSTEXPR` auto **operator()** (`_Tp` &&`__t`, `_Up` &&`__u`) const noexcept(noexcept(`std::forward`<`_Tp`>(`__t`)&&`std::forward`<`_Up`>(`__u`))) -> decltype(`std::forward`<`_Tp`>(`__t`)&&`std::forward`<`_Up`>(`__u`))

5.854.1 Detailed Description

```
template<>
struct std::logical_and< void >
```

One of the [Boolean operations functors](#).

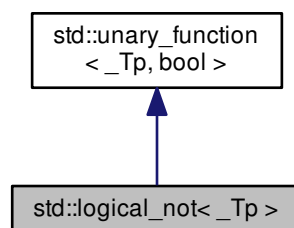
Definition at line 555 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.855 `std::logical_not< _Tp >` Struct Template Reference

Inheritance diagram for `std::logical_not< _Tp >`:



Public Types

- typedef `_Tp` [argument_type](#)
- typedef `bool` [result_type](#)

Public Member Functions

- `_GLIBCXX14_CONSTEXPR` `bool` **operator()** (`const _Tp &__x`) `const`

5.855.1 Detailed Description

```
template<typename _Tp>
struct std::logical_not<_Tp >
```

One of the [Boolean operations functors](#).

Definition at line 519 of file `stl_function.h`.

5.855.2 Member Typedef Documentation

5.855.2.1 `typedef _Tp std::unary_function<_Tp, bool>::argument_type` `[inherited]`

`argument_type` is the type of the argument

Definition at line 108 of file `stl_function.h`.

5.855.2.2 `typedef bool std::unary_function<_Tp, bool>::result_type` `[inherited]`

`result_type` is the return type

Definition at line 111 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.856 `std::logical_not< void >` Struct Template Reference

Public Types

- typedef `__is_transparent` **is_transparent**

Public Member Functions

- `template<typename _Tp >
_GLIBCXX14_CONSTEXPR auto operator() (_Tp &&__t) const noexcept(noexcept(!std::forward< _Tp >(__t)))
-> decltype(!std::forward< _Tp >(__t))`

5.856.1 Detailed Description

```
template<>
struct std::logical_not< void >
```

One of the [Boolean operations functors](#).

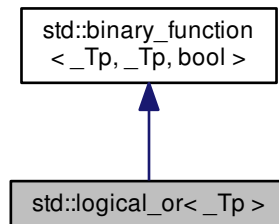
Definition at line 585 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.857 std::logical_or< _Tp > Struct Template Reference

Inheritance diagram for `std::logical_or< _Tp >`:



Public Types

- `typedef _Tp first_argument_type`
- `typedef bool result_type`
- `typedef _Tp second_argument_type`

Public Member Functions

- `_GLIBCXX14_CONSTEXPR bool operator() (const _Tp &__x, const _Tp &__y) const`

5.857.1 Detailed Description

```
template<typename _Tp>
struct std::logical_or< _Tp >
```

One of the [Boolean operations functors](#).

Definition at line 516 of file `stl_function.h`.

5.857.2 Member Typedef Documentation

5.857.2.1 `typedef _Tp std::binary_function< _Tp, _Tp, bool >::first_argument_type` [\[inherited\]](#)

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.857.2.2 `typedef bool std::binary_function< _Tp, _Tp, bool >::result_type` [\[inherited\]](#)

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.857.2.3 `typedef _Tp std::binary_function< _Tp, _Tp, bool >::second_argument_type` [\[inherited\]](#)

`second_argument_type` is the type of the second argument

Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.858 `std::logical_or< void >` Struct Template Reference

Public Types

- `typedef __is_transparent` **is_transparent**

Public Member Functions

- `template<typename _Tp, typename _Up > _GLIBCXX14_CONSTEXPR auto operator() (_Tp &&__t, _Up &&__u) const noexcept(noexcept(std::forward<_Tp>(__t)||std::forward<_Up>(__u))) -> decltype(std::forward<_Tp>(__t)||std::forward<_Up>(__u))`

5.858.1 Detailed Description

```
template<>
struct std::logical_or< void >
```

One of the [Boolean operations functors](#).

Definition at line 570 of file stl_function.h.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.859 std::lognormal_distribution<_RealType> Class Template Reference

Classes

- struct [param_type](#)

Public Types

- typedef _RealType [result_type](#)

Public Member Functions

- **lognormal_distribution** (_RealType __m=_RealType(0), _RealType __s=_RealType(1))
- **lognormal_distribution** (const [param_type](#) &__p)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **__generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **__generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- template<typename _UniformRandomNumberGenerator >
void **__generate** ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- _RealType **m** () const
- [result_type](#) **max** () const
- [result_type](#) **min** () const
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [param_type](#) **param** () const
- void **param** (const [param_type](#) &__param)
- void **reset** ()
- _RealType **s** () const

Friends

- `template<typename _RealType1 , typename _CharT , typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::lognormal_distribution< _RealType1 > &__x)`
- `bool operator== (const lognormal_distribution &__d1, const lognormal_distribution &__d2)`
- `template<typename _RealType1 , typename _CharT , typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, std::`
`::lognormal_distribution< _RealType1 > &__x)`

5.859.1 Detailed Description

```
template<typename _RealType = double>
class std::lognormal_distribution< _RealType >
```

A `lognormal_distribution` random number distribution.

The formula for the normal probability mass function is

$$p(x|m, s) = \frac{1}{sx\sqrt{2\pi}} \exp - \frac{(\ln x - m)^2}{2s^2}$$

Definition at line 2133 of file `random.h`.

5.859.2 Member Typedef Documentation

5.859.2.1 `template<typename _RealType = double> typedef _RealType std::lognormal_distribution< _RealType`
`>::result_type`

The type of the range of the distribution.

Definition at line 2136 of file `random.h`.

5.859.3 Member Function Documentation

5.859.3.1 `template<typename _RealType = double> result_type std::lognormal_distribution< _RealType >::max ()`
`const [inline]`

Returns the least upper bound value of the distribution.

Definition at line 2224 of file `random.h`.

References `std::numeric_limits< _Tp >::max()`.

5.859.3.2 `template<typename _RealType = double> result_type std::lognormal_distribution<_RealType>::min () const`
`[inline]`

Returns the greatest lower bound value of the distribution.

Definition at line 2217 of file random.h.

5.859.3.3 `template<typename _RealType = double> template<typename UniformRandomNumberGenerator> result_type`
`std::lognormal_distribution<_RealType>::operator() (UniformRandomNumberGenerator & __urng)`
`[inline]`

Generating functions.

Definition at line 2232 of file random.h.

References std::exp().

5.859.3.4 `template<typename _RealType = double> param_type std::lognormal_distribution<_RealType>::param ()`
`const [inline]`

Returns the parameter set of the distribution.

Definition at line 2202 of file random.h.

Referenced by std::normal_distribution<_RealType>::operator()().

5.859.3.5 `template<typename _RealType = double> void std::lognormal_distribution<_RealType>::param (const`
`param_type & __param) [inline]`

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

Definition at line 2210 of file random.h.

5.859.3.6 `template<typename _RealType = double> void std::lognormal_distribution<_RealType>::reset ()`
`[inline]`

Resets the distribution state.

Definition at line 2184 of file random.h.

5.859.4 Friends And Related Function Documentation

5.859.4.1 `template<typename _RealType = double> template<typename _RealType1 , typename _CharT , typename _Traits >
std::basic_ostream<_CharT, _Traits>& operator<< (std::basic_ostream< _CharT, _Traits > & __os, const
std::lognormal_distribution<_RealType1 > & __x) [friend]`

Inserts a lognormal_distribution random number distribution __x into the output stream __os.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>lognormal_distribution</code> random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

5.859.4.2 `template<typename _RealType = double> bool operator==(const lognormal_distribution<_RealType> &__d1, const lognormal_distribution<_RealType> &__d2)` [*friend*]

Return true if two `lognormal` distributions have the same parameters and the sequences that would be generated are equal.

Definition at line 2269 of file `random.h`.

5.859.4.3 `template<typename _RealType = double> template<typename _RealType1, typename _CharT, typename _Traits> > std::basic_istream<_CharT, _Traits>& operator>> (std::basic_istream<_CharT, _Traits> &__is, std::lognormal_distribution<_RealType1> &__x)` [*friend*]

Extracts a `lognormal_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>lognormal_distribution</code> random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.860 `std::lognormal_distribution<_RealType>::param_type` Struct Reference

Public Types

- typedef `lognormal_distribution<_RealType>` **distribution_type**

Public Member Functions

- **param_type** (`_RealType __m=_RealType(0), _RealType __s=_RealType(1)`)
- `_RealType m` () const
- `_RealType s` () const

Friends

- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)

5.860.1 Detailed Description

```
template<typename _RealType = double>
struct std::lognormal_distribution< _RealType >::param_type
```

Parameter type.

Definition at line 2142 of file random.h.

The documentation for this struct was generated from the following file:

- [random.h](#)

5.861 `std::map< _Key, _Tp, _Compare, _Alloc >` Class Template Reference

Public Types

- typedef `_Alloc` **allocator_type**
- typedef `_Rep_type::const_iterator` **const_iterator**
- typedef `_Alloc_traits::const_pointer` **const_pointer**
- typedef `_Alloc_traits::const_reference` **const_reference**
- typedef `_Rep_type::const_reverse_iterator` **const_reverse_iterator**
- typedef `_Rep_type::difference_type` **difference_type**
- typedef `_Rep_type::iterator` **iterator**
- typedef `_Compare` **key_compare**
- typedef `_Key` **key_type**
- typedef `_Tp` **mapped_type**
- typedef `_Alloc_traits::pointer` **pointer**
- typedef `_Alloc_traits::reference` **reference**
- typedef `_Rep_type::reverse_iterator` **reverse_iterator**
- typedef `_Rep_type::size_type` **size_type**
- typedef `std::pair< const _Key, _Tp >` **value_type**

Public Member Functions

- [map](#) () noexcept(*/*conditional */*)
- [map](#) (const _Compare &__comp, const allocator_type &__a=allocator_type())
- [map](#) (const [map](#) &__x)
- [map](#) ([map](#) &&__x) noexcept(is_nothrow_copy_constructible< _Compare >::value)
- [map](#) ([initializer_list](#)< [value_type](#) > __l, const _Compare &__comp=_Compare(), const allocator_type &__a=allocator_type())
- [map](#) (const allocator_type &__a)
- [map](#) (const [map](#) &__m, const allocator_type &__a)
- [map](#) ([map](#) &&__m, const allocator_type &__a) noexcept(is_nothrow_copy_constructible< _Compare >::value && _Alloc_traits::_S_always_equal())
- [map](#) ([initializer_list](#)< [value_type](#) > __l, const allocator_type &__a)
- template<typename _InputIterator >
[map](#) (_InputIterator __first, _InputIterator __last, const allocator_type &__a)
- template<typename _InputIterator >
[map](#) (_InputIterator __first, _InputIterator __last)
- template<typename _InputIterator >
[map](#) (_InputIterator __first, _InputIterator __last, const _Compare &__comp, const allocator_type &__a=allocator_type())
- mapped_type & [at](#) (const key_type &__k)
- const mapped_type & [at](#) (const key_type &__k) const
- iterator [begin](#) () noexcept
- const_iterator [begin](#) () const noexcept
- const_iterator [cbegin](#) () const noexcept
- const_iterator [cend](#) () const noexcept
- void [clear](#) () noexcept
- const_reverse_iterator [crbegin](#) () const noexcept
- const_reverse_iterator [crend](#) () const noexcept
- template<typename... _Args>
[std::pair](#)< iterator, bool > [emplace](#) (_Args &&...__args)
- template<typename... _Args>
iterator [emplace_hint](#) (const_iterator __pos, _Args &&...__args)
- bool [empty](#) () const noexcept
- iterator [end](#) () noexcept
- const_iterator [end](#) () const noexcept
- iterator [erase](#) (const_iterator __position)
- _GLIBCXX_ABI_TAG_CXX11 iterator [erase](#) (iterator __position)
- size_type [erase](#) (const key_type &__x)
- iterator [erase](#) (const_iterator __first, const_iterator __last)
- allocator_type [get_allocator](#) () const noexcept
- [std::pair](#)< iterator, bool > [insert](#) (const [value_type](#) &__x)
- template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>
[std::pair](#)< iterator, bool > [insert](#) (_Pair &&__x)
- void [insert](#) ([std::initializer_list](#)< [value_type](#) > __list)
- iterator [insert](#) (const_iterator __position, const [value_type](#) &__x)
- template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>
iterator [insert](#) (const_iterator __position, _Pair &&__x)
- template<typename _InputIterator >
void [insert](#) (_InputIterator __first, _InputIterator __last)
- key_compare [key_comp](#) () const

- `size_type max_size ()` const noexcept
 - `map & operator= (const map &__x)`
 - `map & operator= (map &&)=default`
 - `map & operator= (initializer_list< value_type > __l)`
 - `mapped_type & operator[] (const key_type &__k)`
 - `mapped_type & operator[] (key_type &&__k)`
 - `reverse_iterator rbegin ()` noexcept
 - `const_reverse_iterator rbegin ()` const noexcept
 - `reverse_iterator rend ()` noexcept
 - `const_reverse_iterator rend ()` const noexcept
 - `size_type size ()` const noexcept
 - `void swap (map &__x)` noexcept(/*conditional */)
 - `value_compare value_comp ()` const
-
- iterator `find (const key_type &__x)`
 - `template<typename _Kt >`
`auto find (const _Kt &__x) -> decltype(_M_t._M_find_tr(__x))`
-
- const_iterator `find (const key_type &__x)` const
 - `template<typename _Kt >`
`auto find (const _Kt &__x) const -> decltype(_M_t._M_find_tr(__x))`
-
- `size_type count (const key_type &__x)` const
 - `template<typename _Kt >`
`auto count (const _Kt &__x) const -> decltype(_M_t._M_count_tr(__x))`
-
- iterator `lower_bound (const key_type &__x)`
 - `template<typename _Kt >`
`auto lower_bound (const _Kt &__x) -> decltype(_M_t._M_lower_bound_tr(__x))`
-
- const_iterator `lower_bound (const key_type &__x)` const
 - `template<typename _Kt >`
`auto lower_bound (const _Kt &__x) const -> decltype(_M_t._M_lower_bound_tr(__x))`
-
- iterator `upper_bound (const key_type &__x)`
 - `template<typename _Kt >`
`auto upper_bound (const _Kt &__x) -> decltype(_M_t._M_upper_bound_tr(__x))`
-
- const_iterator `upper_bound (const key_type &__x)` const
 - `template<typename _Kt >`
`auto upper_bound (const _Kt &__x) const -> decltype(_M_t._M_upper_bound_tr(__x))`
-
- `std::pair< iterator, iterator > equal_range (const key_type &__x)`
 - `template<typename _Kt >`
`auto equal_range (const _Kt &__x) -> decltype(_M_t._M_equal_range_tr(__x))`
-
- `std::pair< const_iterator, const_iterator > equal_range (const key_type &__x)` const
 - `template<typename _Kt >`
`auto equal_range (const _Kt &__x) const -> decltype(_M_t._M_equal_range_tr(__x))`

Friends

- `template<typename _K1, typename _T1, typename _C1, typename _A1 >`
`bool operator< (const map<_K1, _T1, _C1, _A1> &, const map<_K1, _T1, _C1, _A1> &)`
- `template<typename _K1, typename _T1, typename _C1, typename _A1 >`
`bool operator== (const map<_K1, _T1, _C1, _A1> &, const map<_K1, _T1, _C1, _A1> &)`

5.861.1 Detailed Description

```
template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<std::pair<const
_Key, _Tp>>>
class std::map<_Key, _Tp, _Compare, _Alloc>
```

A standard container made up of (key,value) pairs, which can be retrieved based on a key, in logarithmic time.

Template Parameters

<code>_Key</code>	Type of key objects.
<code>_Tp</code>	Type of mapped objects.
<code>_Compare</code>	Comparison function object type, defaults to <code>less<_Key></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<pair<const _Key, _Tp>></code> .

Meets the requirements of a [container](#), a [reversible container](#), and an [associative container](#) (using unique keys). For a `map<Key, T>` the `key_type` is `Key`, the `mapped_type` is `T`, and the `value_type` is `std::pair<const Key, T>`.

Maps support bidirectional iterators.

The private tree data is declared exactly the same way for `map` and `multimap`; the distinction is made entirely in how the tree functions are called (`*_unique` versus `*_equal`, same as the standard).

Definition at line 96 of file `stl_map.h`.

5.861.2 Constructor & Destructor Documentation

5.861.2.1 `template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<std::pair<const _Key, _Tp>>> std::map<_Key, _Tp, _Compare, _Alloc>::map () [inline], [noexcept]`

Default constructor creates no elements.

Definition at line 162 of file `stl_map.h`.

5.861.2.2 `template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<std::pair<const _Key, _Tp>>> std::map<_Key, _Tp, _Compare, _Alloc>::map (const _Compare & __comp, const allocator_type & __a = allocator_type()) [inline], [explicit]`

Creates a map with no elements.

Parameters

<code>__comp</code>	A comparison object.
<code>__a</code>	An allocator object.

Definition at line 174 of file `stl_map.h`.

```
5.861.2.3  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> std::map<_Key, _Tp, _Compare, _Alloc>::map ( const map<
_Key, _Tp, _Compare, _Alloc> &__x )  [inline]
```

Map copy constructor.

Parameters

<code>__x</code>	A map of identical element and allocator types.
------------------	---

The newly-created map uses a copy of the allocation object used by `__x`.

Definition at line 185 of file `stl_map.h`.

```
5.861.2.4  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> std::map<_Key, _Tp, _Compare, _Alloc>::map ( map<_Key, _Tp,
_Compare, _Alloc> &&__x )  [inline], [noexcept]
```

Map move constructor.

Parameters

<code>__x</code>	A map of identical element and allocator types.
------------------	---

The newly-created map contains the exact contents of `__x`. The contents of `__x` are a valid, but unspecified map.

Definition at line 196 of file `stl_map.h`.

```
5.861.2.5  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> std::map<_Key, _Tp, _Compare, _Alloc>::map (
initializer_list<value_type> &__l, const _Compare &__comp = _Compare(), const allocator_type &__a =
allocator_type() )  [inline]
```

Builds a map from an `initializer_list`.

Parameters

<code>__l</code>	An <code>initializer_list</code> .
<code>__comp</code>	A comparison object.
<code>__a</code>	An allocator object.

Create a map consisting of copies of the elements in the initializer_list __l. This is linear in N if the range is already sorted, and NlogN otherwise (where N is __l.size()).

Definition at line 211 of file stl_map.h.

```
5.861.2.6 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> std::map<_Key, _Tp, _Compare, _Alloc>::map ( const
allocator_type & __a ) [inline], [explicit]
```

Allocator-extended default constructor.

Definition at line 219 of file stl_map.h.

```
5.861.2.7 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> std::map<_Key, _Tp, _Compare, _Alloc>::map ( const map<
_Key, _Tp, _Compare, _Alloc> & __m, const allocator_type & __a ) [inline]
```

Allocator-extended copy constructor.

Definition at line 223 of file stl_map.h.

```
5.861.2.8 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> std::map<_Key, _Tp, _Compare, _Alloc>::map ( map<_Key, _Tp,
_Compare, _Alloc> && __m, const allocator_type & __a ) [inline], [noexcept]
```

Allocator-extended move constructor.

Definition at line 227 of file stl_map.h.

```
5.861.2.9 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> std::map<_Key, _Tp, _Compare, _Alloc>::map ( initializer_list<
value_type> __l, const allocator_type & __a ) [inline]
```

Allocator-extended initialier-list constructor.

Definition at line 233 of file stl_map.h.

```
5.861.2.10 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> template<typename _InputIterator> std::map<_Key, _Tp,
_Compare, _Alloc>::map ( _InputIterator __first, _InputIterator __last, const allocator_type & __a ) [inline]
```

Allocator-extended range constructor.

Definition at line 239 of file stl_map.h.

```
5.861.2.11 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> template<typename _InputIterator> std::map<_Key, _Tp,
_Compare, _Alloc>::map ( _InputIterator __first, _InputIterator __last ) [inline]
```

Builds a map from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Create a map consisting of copies of the elements from `[__first,__last)`. This is linear in N if the range is already sorted, and $N\log N$ otherwise (where N is `distance(__first,__last)`).

Definition at line 256 of file `stl_map.h`.

```
5.861.2.12  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
             std::allocator<std::pair<const _Key, _Tp>>>> template<typename _InputIterator> std::map<_Key, _Tp,
             _Compare, _Alloc>::map ( _InputIterator __first, _InputIterator __last, const _Compare & __comp, const
             allocator_type & __a = allocator_type() ) [inline]
```

Builds a map from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__comp</code>	A comparison functor.
<code>__a</code>	An allocator object.

Create a map consisting of copies of the elements from `[__first,__last)`. This is linear in N if the range is already sorted, and $N\log N$ otherwise (where N is `distance(__first,__last)`).

Definition at line 273 of file `stl_map.h`.

5.861.3 Member Function Documentation

```
5.861.3.1  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
             std::allocator<std::pair<const _Key, _Tp>>>> mapped_type& std::map<_Key, _Tp, _Compare, _Alloc>::at ( const
             key_type & __k ) [inline]
```

Access to map data.

Parameters

<code>__k</code>	The key for which data should be retrieved.
------------------	---

Returns

A reference to the data whose key is equivalent to `__k`, if such a data is present in the map.

Exceptions

<code>std::out_of_range</code>	If no such data is present.
--------------------------------	-----------------------------

Definition at line 519 of file `stl_map.h`.

References `std::map< _Key, _Tp, _Compare, _Alloc >::end()`, `std::map< _Key, _Tp, _Compare, _Alloc >::key_comp()`, and `std::map< _Key, _Tp, _Compare, _Alloc >::lower_bound()`.

5.861.3.2 `template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<std::pair<const _Key, _Tp>>> iterator std::map< _Key, _Tp, _Compare, _Alloc >::begin ()`
`[inline], [noexcept]`

Returns a read/write iterator that points to the first pair in the map. Iteration is done in ascending order according to the keys.

Definition at line 338 of file `stl_map.h`.

5.861.3.3 `template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<std::pair<const _Key, _Tp>>> const_iterator std::map< _Key, _Tp, _Compare, _Alloc >::begin ()`
`const [inline], [noexcept]`

Returns a read-only (constant) iterator that points to the first pair in the map. Iteration is done in ascending order according to the keys.

Definition at line 347 of file `stl_map.h`.

5.861.3.4 `template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<std::pair<const _Key, _Tp>>> const_iterator std::map< _Key, _Tp, _Compare, _Alloc >::cbegin ()`
`const [inline], [noexcept]`

Returns a read-only (constant) iterator that points to the first pair in the map. Iteration is done in ascending order according to the keys.

Definition at line 411 of file `stl_map.h`.

5.861.3.5 `template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<std::pair<const _Key, _Tp>>> const_iterator std::map< _Key, _Tp, _Compare, _Alloc >::cend ()`
`const [inline], [noexcept]`

Returns a read-only (constant) iterator that points one past the last pair in the map. Iteration is done in ascending order according to the keys.

Definition at line 420 of file `stl_map.h`.

5.861.3.6 `template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<std::pair<const _Key, _Tp>>> void std::map< _Key, _Tp, _Compare, _Alloc >::clear ()`
`[inline], [noexcept]`

Erases all elements in a map. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1043 of file `stl_map.h`.

5.861.3.7 `template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<std::pair<const _Key, _Tp>>> size_type std::map< _Key, _Tp, _Compare, _Alloc >::count (const key_type & __x) const` `[inline]`

Finds the number of elements with given key.

Parameters

<code>_↔</code>	Key of (key, value) pairs to be located.
<code>_x</code>	

Returns

Number of elements with specified key.

This function only makes sense for multimaps; for map the result will either be 0 (not present) or 1 (present).

Definition at line 1125 of file `stl_map.h`.

```
5.861.3.8  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
           std::allocator<std::pair<const _Key, _Tp> >> template<typename _Kt > auto std::map<_Key, _Tp, _Compare,
           _Alloc>::count ( const _Kt & _x ) const -> decltype(_M_t._M_count_tr(_x))  [inline]
```

Finds the number of elements with given key.

Parameters

<code>_↔</code>	Key of (key, value) pairs to be located.
<code>_x</code>	

Returns

Number of elements with specified key.

This function only makes sense for multimaps; for map the result will either be 0 (not present) or 1 (present).

Definition at line 1131 of file `stl_map.h`.

```
5.861.3.9  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
           std::allocator<std::pair<const _Key, _Tp> >> const_reverse_iterator std::map<_Key, _Tp, _Compare, _Alloc
           >::rbegin ( ) const  [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last pair in the map. Iteration is done in descending order according to the keys.

Definition at line 429 of file `stl_map.h`.

```
5.861.3.10 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
           std::allocator<std::pair<const _Key, _Tp> >> const_reverse_iterator std::map<_Key, _Tp, _Compare, _Alloc
           >::crend ( ) const  [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to one before the first pair in the map. Iteration is done in descending order according to the keys.

Definition at line 438 of file `stl_map.h`.

```
5.861.3.11 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =  
std::allocator<std::pair<const _Key, _Tp> >> template<typename... _Args> std::pair<iterator, bool>  
std::map<_Key, _Tp, _Compare, _Alloc >::emplace ( _Args &&... __args ) [inline]
```

Attempts to build and insert a std::pair into the map.

Parameters

<code>__args</code>	Arguments used to generate a new pair instance (see <code>std::piecewise_construct</code> for passing arguments to each part of the pair constructor).
---------------------	--

Returns

A pair, of which the first element is an iterator that points to the possibly inserted pair, and the second is a bool that is true if the pair was actually inserted.

This function attempts to build and insert a (key, value) pair into the map. A map relies on unique keys and thus a pair is only inserted if its first element (the key) is not already present in the map.

Insertion requires logarithmic time.

Definition at line 558 of file `stl_map.h`.

```
5.861.3.12  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp> >> template<typename... _Args> iterator std::map<_Key, _Tp,
            _Compare, _Alloc >::emplace_hint ( const_iterator __pos, _Args &&... __args ) [inline]
```

Attempts to build and insert a `std::pair` into the map.

Parameters

<code>__pos</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__args</code>	Arguments used to generate a new pair instance (see <code>std::piecewise_construct</code> for passing arguments to each part of the pair constructor).

Returns

An iterator that points to the element with key of the `std::pair` built from `__args` (may or may not be that `std::pair`).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `emplace()` does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints for more on *hinting*.

Insertion requires logarithmic time (if the hint is not taken).

Definition at line 588 of file `stl_map.h`.

References `std::map<_Key, _Tp, _Compare, _Alloc >::end()`, `std::map<_Key, _Tp, _Compare, _Alloc >::key_comp()`, `std::map<_Key, _Tp, _Compare, _Alloc >::lower_bound()`, and `std::piecewise_construct`.

Referenced by `std::map<_Key, _Tp, _Compare, _Alloc >::insert()`.

```
5.861.3.13 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> bool std::map<_Key, _Tp, _Compare, _Alloc>::empty ( ) const
[inline], [noexcept]
```

Returns true if the map is empty. (Thus begin() would equal end().)

Definition at line 447 of file stl_map.h.

```
5.861.3.14 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> iterator std::map<_Key, _Tp, _Compare, _Alloc>::end ( )
[inline], [noexcept]
```

Returns a read/write iterator that points one past the last pair in the map. Iteration is done in ascending order according to the keys.

Definition at line 356 of file stl_map.h.

Referenced by std::map< _Key, _Tp, _Compare, _Alloc>::at(), std::map< _Key, _Tp, _Compare, _Alloc>::emplace_↔ hint(), std::map< _Key, _Tp, _Compare, _Alloc>::insert(), and std::map< _Key, _Tp, _Compare, _Alloc>::operator[]().

```
5.861.3.15 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> const_iterator std::map<_Key, _Tp, _Compare, _Alloc>::end ( )
const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last pair in the map. Iteration is done in ascending order according to the keys.

Definition at line 365 of file stl_map.h.

```
5.861.3.16 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> std::pair<iterator, iterator> std::map<_Key, _Tp, _Compare,
_Alloc>::equal_range ( const key_type & __x ) [inline]
```

Finds a subsequence matching given key.

Parameters

__↔ __x	Key of (key, value) pairs to be located.
--------------------	--

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multimaps.

Definition at line 1243 of file stl_map.h.

```
5.861.3.17 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> template<typename _Kt > auto std::map<_Key, _Tp, _Compare,
_Alloc >::equal_range ( const _Kt & __x ) -> decltype(_M.t._M_equal_range_tr(__x)) [inline]
```

Finds a subsequence matching given key.

Parameters

\leftrightarrow	Key of (key, value) pairs to be located.
x	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multimaps.

Definition at line 1249 of file stl_map.h.

```
5.861.3.18 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> std::pair<const_iterator, const_iterator> std::map<_Key, _Tp,
_Compare, _Alloc >::equal_range ( const key_type & __x ) const [inline]
```

Finds a subsequence matching given key.

Parameters

\leftrightarrow	Key of (key, value) pairs to be located.
x	

Returns

Pair of read-only (constant) iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
               c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multimaps.

Definition at line 1272 of file stl_map.h.

```
5.861.3.19 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> template<typename _Kt > auto std::map<_Key, _Tp, _Compare,
_Alloc>::equal_range ( const _Kt & __x ) const -> decltype(_M_t._M_equal_range_tr(__x)) [inline]
```

Finds a subsequence matching given key.

Parameters

<code>__x</code>	Key of (key, value) pairs to be located.
------------------	--

Returns

Pair of read-only (constant) iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
               c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multimaps.

Definition at line 1278 of file stl_map.h.

References `std::operator==()`.

```
5.861.3.20 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> iterator std::map<_Key, _Tp, _Compare, _Alloc>::erase (
const_iterator __position ) [inline]
```

Erases an element from a map.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following *position* prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from a map. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 944 of file `stl_map.h`.

Referenced by `std::map<_Key, _Tp, _Compare, _Alloc>::erase()`.

```
5.861.3.21  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp>>> size_type std::map<_Key, _Tp, _Compare, _Alloc>::erase ( const
            key_type & __x )  [inline]
```

Erases elements according to the provided key.

Parameters

<code>__x</code>	Key of element to be erased.
------------------	------------------------------

Returns

The number of elements erased.

This function erases all the elements located by the given key from a map. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 980 of file `stl_map.h`.

```
5.861.3.22  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp>>> iterator std::map<_Key, _Tp, _Compare, _Alloc>::erase (
            const_iterator __first, const_iterator __last )  [inline]
```

Erases a `[first,last)` range of elements from a map.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be erased.
<code>__last</code>	Iterator pointing to the end of the range to be erased.

Returns

The iterator `__last`.

This function erases a sequence of elements from a map. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1000 of file stl_map.h.

References `std::map< _Key, _Tp, _Compare, _Alloc >::erase()`.

5.861.3.23 `template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<std::pair<const _Key, _Tp>>> iterator std::map< _Key, _Tp, _Compare, _Alloc >::find (const key_type & __x) [inline]`

Tries to locate an element in a map.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Iterator pointing to sought-after element, or `end()` if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after pair. If unsuccessful it returns the past-the-end (`end()`) iterator.

Definition at line 1079 of file stl_map.h.

5.861.3.24 `template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<std::pair<const _Key, _Tp>>> template<typename _Kt > auto std::map< _Key, _Tp, _Compare, _Alloc >::find (const _Kt & __x)-> decltype(_M_t._M_find_tr(__x)) [inline]`

Tries to locate an element in a map.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Iterator pointing to sought-after element, or `end()` if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after pair. If unsuccessful it returns the past-the-end (`end()`) iterator.

Definition at line 1085 of file stl_map.h.

```
5.861.3.25 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> const_iterator std::map<_Key, _Tp, _Compare, _Alloc >::find (
const key_type & __x ) const [inline]
```

Tries to locate an element in a map.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Read-only (constant) iterator pointing to sought-after element, or `end()` if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns a constant iterator pointing to the sought after pair. If unsuccessful it returns the past-the-end (`end()`) iterator.

Definition at line 1104 of file `stl_map.h`.

```
5.861.3.26 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> template<typename _Kt > auto std::map<_Key, _Tp, _Compare,
_Alloc >::find ( const _Kt & __x ) const -> decltype(_M_t._M_find_tr(__x)) [inline]
```

Tries to locate an element in a map.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Read-only (constant) iterator pointing to sought-after element, or `end()` if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns a constant iterator pointing to the sought after pair. If unsuccessful it returns the past-the-end (`end()`) iterator.

Definition at line 1110 of file `stl_map.h`.

```
5.861.3.27 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> allocator_type std::map<_Key, _Tp, _Compare, _Alloc
>::get_allocator ( ) const [inline], [noexcept]
```

Get a copy of the memory allocation object.

Definition at line 328 of file `stl_map.h`.

```
5.861.3.28 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =  
std::allocator<std::pair<const _Key, _Tp>>> std::pair<iterator, bool> std::map<_Key, _Tp, _Compare, _Alloc  
>::insert ( const value_type & __x ) [inline]
```

Attempts to insert a std::pair into the map.

Parameters

<code>_↔</code>	Pair to be inserted (see <code>std::make_pair</code> for easy creation of pairs).
<code>_x</code>	

Returns

A pair, of which the first element is an iterator that points to the possibly inserted pair, and the second is a bool that is true if the pair was actually inserted.

This function attempts to insert a (key, value) pair into the map. A map relies on unique keys and thus a pair is only inserted if its first element (the key) is not already present in the map.

Insertion requires logarithmic time.

Definition at line 731 of file `stl_map.h`.

Referenced by `std::map<_Key, _Tp, _Compare, _Alloc>::insert()`, and `std::map<_Key, _Tp, _Compare, _Alloc>::operator[]()`.

```
5.861.3.29  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp>>> void std::map<_Key, _Tp, _Compare, _Alloc>::insert (
            std::initializer_list<value_type> &__list ) [inline]
```

Attempts to insert a list of `std::pairs` into the map.

Parameters

<code>__list</code>	A <code>std::initializer_list<value_type></code> of pairs to be inserted.
---------------------	---

Complexity similar to that of the range constructor.

Definition at line 752 of file `stl_map.h`.

References `std::map<_Key, _Tp, _Compare, _Alloc>::insert()`.

```
5.861.3.30  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp>>> iterator std::map<_Key, _Tp, _Compare, _Alloc>::insert (
            const_iterator __position, const value_type &__x ) [inline]
```

Attempts to insert a `std::pair` into the map.

Parameters

<code>__position</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__x</code>	Pair to be inserted (see <code>std::make_pair</code> for easy creation of pairs).

Returns

An iterator that points to the element with key of `__x` (may or may not be the pair passed in).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `insert()` does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints for more on *hinting*.

Insertion requires logarithmic time (if the hint is not taken).

Definition at line 781 of file `stl_map.h`.

References `std::map< _Key, _Tp, _Compare, _Alloc >::insert()`.

```
5.861.3.31  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp> >> template<typename _InputIterator > void std::map< _Key, _Tp,
            _Compare, _Alloc >::insert ( _InputIterator __first, _InputIterator __last ) [inline]
```

Template function that attempts to insert a range of elements.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be inserted.
<code>__last</code>	Iterator pointing to the end of the range.

Complexity similar to that of the range constructor.

Definition at line 807 of file `stl_map.h`.

References `std::map< _Key, _Tp, _Compare, _Alloc >::emplace_hint()`, `std::map< _Key, _Tp, _Compare, _Alloc >::end()`, `std::map< _Key, _Tp, _Compare, _Alloc >::key_comp()`, `std::map< _Key, _Tp, _Compare, _Alloc >::lower_bound()`, and `std::piecewise_construct`.

```
5.861.3.32  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp> >> key_compare std::map< _Key, _Tp, _Compare, _Alloc >::key_comp (
            ) const [inline]
```

Returns the key comparison object out of which the map was constructed.

Definition at line 1052 of file `stl_map.h`.

Referenced by `std::map< _Key, _Tp, _Compare, _Alloc >::at()`, `std::map< _Key, _Tp, _Compare, _Alloc >::emplace_hint()`, `std::map< _Key, _Tp, _Compare, _Alloc >::insert()`, and `std::map< _Key, _Tp, _Compare, _Alloc >::operator[]()`.

```
5.861.3.33  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp> >> iterator std::map< _Key, _Tp, _Compare, _Alloc >::lower_bound (
            const key_type & __x ) [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

<code>_↔</code>	Key of (key, value) pair to be located.
<code>_X</code>	

Returns

Iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

Definition at line 1149 of file `stl_map.h`.

Referenced by `std::map<_Key, _Tp, _Compare, _Alloc>::at()`, `std::map<_Key, _Tp, _Compare, _Alloc>::emplace_↔hint()`, `std::map<_Key, _Tp, _Compare, _Alloc>::insert()`, and `std::map<_Key, _Tp, _Compare, _Alloc>::operator[]()`.

```
5.861.3.34  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp>>> template<typename _Kt > auto std::map<_Key, _Tp, _Compare,
            _Alloc>::lower_bound ( const _Kt & __x ) -> decltype(_M.t._M_lower_bound_tr(__x))  [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

<code>_↔</code>	Key of (key, value) pair to be located.
<code>_X</code>	

Returns

Iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

Definition at line 1155 of file `stl_map.h`.

```
5.861.3.35  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp>>> const_iterator std::map<_Key, _Tp, _Compare, _Alloc
            >::lower_bound ( const key_type & __x ) const  [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

<code>_↔</code>	Key of (key, value) pair to be located.
<code>_X</code>	

Returns

Read-only (constant) iterator pointing to first element equal to or greater than key, or end().



This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

Definition at line 1174 of file stl_map.h.

```
5.861.3.36 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> template<typename _Kt > auto std::map<_Key, _Tp, _Compare,
_Alloc >::lower_bound ( const _Kt & __x ) const -> decltype(_M_t._M_lower_bound_tr(__x)) [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

	Key of (key, value) pair to be located.
	

Returns

Read-only (constant) iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

Definition at line 1180 of file stl_map.h.

```
5.861.3.37 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> size_type std::map<_Key, _Tp, _Compare, _Alloc >::max_size ( )
const [inline], [noexcept]
```

Returns the maximum size of the map.



Definition at line 457 of file stl_map.h.

```
5.861.3.38 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> map& std::map<_Key, _Tp, _Compare, _Alloc >::operator= ( const
map<_Key, _Tp, _Compare, _Alloc > & __x ) [inline]
```

Map assignment operator.

The dtor only erases the elements, and note that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Parameters

	A map of identical element and allocator types.
	

All the elements of `__x` are copied, but unlike the copy constructor, the allocator object is not copied.

Definition at line 296 of file `stl_map.h`.

```
5.861.3.39  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp>>> map& std::map<_Key, _Tp, _Compare, _Alloc>::operator= (
            map<_Key, _Tp, _Compare, _Alloc> && ) [default]
```

Move assignment operator.

```
5.861.3.40  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp>>> map& std::map<_Key, _Tp, _Compare, _Alloc>::operator= (
            initializer_list<value_type> __l ) [inline]
```

Map list assignment operator.

Parameters

↔	An <code>initializer_list</code> .
↔	
↔	
↔	
<code>l</code>	

This function fills a map with copies of the elements in the initializer list `__l`.

Note that the assignment completely changes the map and that the resulting map's size is the same as the number of elements assigned. Old data may be lost.

Definition at line 319 of file `stl_map.h`.

```
5.861.3.41  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp>>> mapped_type& std::map<_Key, _Tp, _Compare, _Alloc>
            >::operator[] ( const key_type & __k ) [inline]
```

Subscript (`[]`) access to map data.

Parameters

↔	The key for which data should be retrieved.
<code>k</code>	

Returns

A reference to the data of the (key,data) pair.

Allows for easy lookup with the subscript (`[]`) operator. Returns data associated with the key specified in subscript. If the key does not exist, a pair with that key is created using default values, which is then returned.

Lookup requires logarithmic time.

Definition at line 474 of file stl_map.h.

References std::map< _Key, _Tp, _Compare, _Alloc >::end(), std::map< _Key, _Tp, _Compare, _Alloc >::insert(), std::map< _Key, _Tp, _Compare, _Alloc >::key_comp(), std::map< _Key, _Tp, _Compare, _Alloc >::lower_bound(), and std::piecewise_construct.

```
5.861.3.42 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> reverse_iterator std::map< _Key, _Tp, _Compare, _Alloc
>::rbegin( ) [inline], [noexcept]
```

Returns a read/write reverse iterator that points to the last pair in the map. Iteration is done in descending order according to the keys.

Definition at line 374 of file stl_map.h.

```
5.861.3.43 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> const_reverse_iterator std::map< _Key, _Tp, _Compare, _Alloc
>::rbegin( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last pair in the map. Iteration is done in descending order according to the keys.

Definition at line 383 of file stl_map.h.

```
5.861.3.44 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> reverse_iterator std::map< _Key, _Tp, _Compare, _Alloc >::rend(
) [inline], [noexcept]
```

Returns a read/write reverse iterator that points to one before the first pair in the map. Iteration is done in descending order according to the keys.

Definition at line 392 of file stl_map.h.

```
5.861.3.45 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> const_reverse_iterator std::map< _Key, _Tp, _Compare, _Alloc
>::rend( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to one before the first pair in the map. Iteration is done in descending order according to the keys.

Definition at line 401 of file stl_map.h.

```
5.861.3.46 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> size_type std::map< _Key, _Tp, _Compare, _Alloc >::size( ) const
[inline], [noexcept]
```

Returns the size of the map.

Definition at line 452 of file stl_map.h.

```
5.861.3.47 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> void std::map< _Key, _Tp, _Compare, _Alloc >::swap( map<
_Key, _Tp, _Compare, _Alloc > &_x ) [inline], [noexcept]
```

Swaps data with another map.

Parameters

<code>_↔</code>	A map of the same element and allocator types.
<code>_X</code>	

This exchanges the elements between two maps in constant time. (It is only swapping a pointer, an integer, and an instance of the `Compare` type (which itself is often stateless and empty), so it should be quite fast.) Note that the global `std::swap()` function is specialized such that `std::swap(m1,m2)` will feed to this function.

Definition at line 1032 of file `stl_map.h`.

```
5.861.3.48  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp> >> iterator std::map<_Key, _Tp, _Compare, _Alloc >::upper_bound (
            const key_type & __x )  [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>_↔</code>	Key of (key, value) pair to be located.
<code>_X</code>	

Returns

Iterator pointing to the first element greater than key, or `end()`.

Definition at line 1194 of file `stl_map.h`.

```
5.861.3.49  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp> >> template<typename _Kt > auto std::map<_Key, _Tp, _Compare,
            _Alloc >::upper_bound ( const _Kt & __x ) -> decltype(_M_t._M_upper_bound_tr(__x))  [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>_↔</code>	Key of (key, value) pair to be located.
<code>_X</code>	

Returns

Iterator pointing to the first element greater than key, or `end()`.

Definition at line 1200 of file `stl_map.h`.

```
5.861.3.50 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >> const_iterator std::map< _Key, _Tp, _Compare, _Alloc
>::upper_bound ( const key_type & __x ) const [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Read-only (constant) iterator pointing to first iterator greater than key, or end().

Definition at line 1214 of file stl_map.h.

```
5.861.3.51 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >> template<typename _Kt > auto std::map< _Key, _Tp, _Compare,
_Alloc >::upper_bound ( const _Kt & __x ) const -> decltype(_M_t._M_upper_bound_tr(__x)) [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Read-only (constant) iterator pointing to first iterator greater than key, or end().

Definition at line 1220 of file stl_map.h.

```
5.861.3.52 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >> value_compare std::map< _Key, _Tp, _Compare, _Alloc
>::value_comp ( ) const [inline]
```

Returns a value comparison object, built from the key comparison object out of which the map was constructed.

Definition at line 1060 of file stl_map.h.

The documentation for this class was generated from the following file:

- [stl_map.h](#)

5.862 `std::mask_array<_Tp>` Class Template Reference

Public Types

- typedef `_Tp` **value_type**

Public Member Functions

- `mask_array` (const `mask_array` &)
- void `operator%=(const valarray<_Tp> &) const`
- template<class `_Dom` >
void `operator%=(const _Expr<_Dom, _Tp> &) const`
- void `operator&=(const valarray<_Tp> &) const`
- template<class `_Dom` >
void `operator&=(const _Expr<_Dom, _Tp> &) const`
- void `operator*=(const valarray<_Tp> &) const`
- template<class `_Dom` >
void `operator*=(const _Expr<_Dom, _Tp> &) const`
- void `operator+=(const valarray<_Tp> &) const`
- template<class `_Dom` >
void `operator+=(const _Expr<_Dom, _Tp> &) const`
- void `operator-=(const valarray<_Tp> &) const`
- template<class `_Dom` >
void `operator-=(const _Expr<_Dom, _Tp> &) const`
- void `operator/=(const valarray<_Tp> &) const`
- template<class `_Dom` >
void `operator/=(const _Expr<_Dom, _Tp> &) const`
- void `operator<=<=` (const `valarray<_Tp> &) const`
- template<class `_Dom` >
void `operator<=<=` (const `_Expr<_Dom, _Tp> &) const`
- `mask_array` & `operator=` (const `mask_array` &)
- void `operator=` (const `valarray<_Tp> &) const`
- void `operator=` (const `_Tp` &) const
- template<class `_Dom` >
void `operator=` (const `_Expr<_Dom, _Tp> &) const`
- template<class `_Ex` >
void `operator=` (const `_Expr<_Ex, _Tp> &__e`) const
- void `operator>>=` (const `valarray<_Tp> &) const`
- template<class `_Dom` >
void `operator>>=` (const `_Expr<_Dom, _Tp> &) const`
- void `operator^=` (const `valarray<_Tp> &) const`
- template<class `_Dom` >
void `operator^=` (const `_Expr<_Dom, _Tp> &) const`
- void `operator|=` (const `valarray<_Tp> &) const`
- template<class `_Dom` >
void `operator|=` (const `_Expr<_Dom, _Tp> &) const`

Friends

- class `valarray<_Tp>`

5.862.1 Detailed Description

```
template<class _Tp>
class std::mask_array<_Tp>
```

Reference to selected subset of an array.

A mask_array is a reference to the actual elements of an array specified by a bitmask in the form of an array of bool. The way to get a mask_array is to call operator[](valarray<bool>) on a valarray. The returned mask_array then permits carrying operations out on the referenced subset of elements in the original valarray.

For example, if a mask_array is obtained using the array (false, true, false, true) as an argument, the mask array has two elements referring to array[1] and array[3] in the underlying array.

Parameters

<i>Tp</i>	Element type.
-----------	---------------

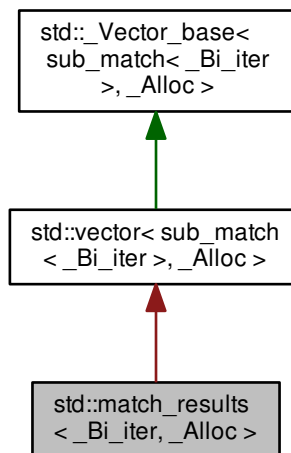
Definition at line 83 of file valarray.

The documentation for this class was generated from the following files:

- [valarray](#)
- [mask_array.h](#)

5.863 std::match_results<_Bi_iter, _Alloc> Class Template Reference

Inheritance diagram for std::match_results<_Bi_iter, _Alloc>:



Public Member Functions

- `template<typename _Out_iter >`
`_Out_iter format (_Out_iter __out, const match_results< _Bi_iter, _Alloc >::char_type *__fmt_first, const`
`match_results< _Bi_iter, _Alloc >::char_type *__fmt_last, match_flag_type __flags) const`
- `bool ready () const`

Private Types

- `typedef _Alloc_traits::const_pointer const_pointer`
- `typedef std::reverse_iterator< const_iterator > const_reverse_iterator`
- `typedef _Base::pointer pointer`
- `typedef std::reverse_iterator< iterator > reverse_iterator`

Private Member Functions

- `pointer _M_allocate (size_t __n)`
- `pointer _M_allocate_and_copy (size_type __n, _ForwardIterator __first, _ForwardIterator __last)`
- `void _M_assign_aux (_InputIterator __first, _InputIterator __last, std::input_iterator_tag)`
- `void _M_assign_aux (_ForwardIterator __first, _ForwardIterator __last, std::forward_iterator_tag)`
- `void _M_assign_dispatch (_Integer __n, _Integer __val, __true_type)`
- `void _M_assign_dispatch (_InputIterator __first, _InputIterator __last, __false_type)`
- `size_type _M_check_len (size_type __n, const char *__s) const`
- `void _M_deallocate (pointer __p, size_t __n)`
- `void _M_default_append (size_type __n)`
- `void _M_default_initialize (size_type __n)`
- `void _M_emplace_back_aux (_Args &&...__args)`
- `iterator _M_erase (iterator __position)`
- `iterator _M_erase (iterator __first, iterator __last)`
- `void _M_erase_at_end (pointer __pos) noexcept`
- `void _M_fill_assign (size_type __n, const value_type &__val)`
- `void _M_fill_initialize (size_type __n, const value_type &__value)`
- `void _M_fill_insert (iterator __pos, size_type __n, const value_type &__x)`
- `_Tp_alloc_type & _M_get_Tp_allocator () noexcept`
- `const _Tp_alloc_type & _M_get_Tp_allocator () const noexcept`
- `void _M_initialize_dispatch (_Integer __n, _Integer __value, __true_type)`
- `void _M_initialize_dispatch (_InputIterator __first, _InputIterator __last, __false_type)`
- `void _M_insert_aux (iterator __position, _Args &&...__args)`
- `void _M_insert_dispatch (iterator __pos, _Integer __n, _Integer __val, __true_type)`
- `void _M_insert_dispatch (iterator __pos, _InputIterator __first, _InputIterator __last, __false_type)`
- `void _M_range_check (size_type __n) const`
- `void _M_range_initialize (_InputIterator __first, _InputIterator __last, std::input_iterator_tag)`
- `void _M_range_initialize (_ForwardIterator __first, _ForwardIterator __last, std::forward_iterator_tag)`
- `void _M_range_insert (iterator __pos, _InputIterator __first, _InputIterator __last, std::input_iterator_tag)`
- `void _M_range_insert (iterator __pos, _ForwardIterator __first, _ForwardIterator __last, std::forward_iterator_tag)`
- `bool _M_shrink_to_fit ()`
- `void assign (size_type __n, const value_type &__val)`
- `void assign (_InputIterator __first, _InputIterator __last)`

- void `assign` (initializer_list< value_type > __l)
- `reference` at (size_type __n)
- `const_reference` at (size_type __n) const
- `reference` back () noexcept
- `const_reference` back () const noexcept
- iterator `begin` () noexcept
- size_type `capacity` () const noexcept
- void `clear` () noexcept
- `const_reverse_iterator` `crbegin` () const noexcept
- `const_reverse_iterator` `crend` () const noexcept
- `sub_match`< _Bi_iter > * `data` () noexcept
- const `sub_match`< _Bi_iter > * `data` () const noexcept
- iterator `emplace` (const_iterator __position, _Args &&... __args)
- void `emplace_back` (_Args &&... __args)
- iterator `end` () noexcept
- iterator `erase` (const_iterator __position)
- iterator `erase` (const_iterator __first, const_iterator __last)
- `reference` front () noexcept
- `const_reference` front () const noexcept
- iterator `insert` (const_iterator __position, const value_type & __x)
- iterator `insert` (const_iterator __position, value_type && __x)
- iterator `insert` (const_iterator __position, initializer_list< value_type > __l)
- iterator `insert` (const_iterator __position, size_type __n, const value_type & __x)
- iterator `insert` (const_iterator __position, _InputIterator __first, _InputIterator __last)
- `reference` operator[] (size_type __n) noexcept
- `const_reference` operator[] (size_type __n) const noexcept
- void `pop_back` () noexcept
- void `push_back` (const value_type & __x)
- void `push_back` (value_type && __x)
- `reverse_iterator` `rbegin` () noexcept
- `const_reverse_iterator` `rbegin` () const noexcept
- `reverse_iterator` `rend` () noexcept
- `const_reverse_iterator` `rend` () const noexcept
- void `reserve` (size_type __n)
- void `resize` (size_type __new_size)
- void `resize` (size_type __new_size, const value_type & __x)
- void `shrink_to_fit` ()
- void `swap` (vector & __x) noexcept

Private Attributes

- `_Vector_impl _M_impl`

Friends

- template<typename _Bp, typename _Ap, typename _Cp, typename _Rp, __detail::__RegexExecutorPolicy, bool >
bool `__detail::__regex_algo_impl` (_Bp, _Bp, match_results< _Bp, _Ap > &, const basic_regex< _Cp, _Rp > &, regex_constants::match_flag_type)
- template<typename, typename, typename, bool >
class `__detail::__Executor`
- template<typename, typename, typename >
class `regex_iterator`

10.? Public Types

- typedef [sub_match](#)< [_Bi_iter](#) > **value_type**
- typedef const [value_type](#) & **const_reference**
- typedef [const_reference](#) **reference**
- typedef [_Base_type::const_iterator](#) **const_iterator**
- typedef [const_iterator](#) **iterator**
- typedef [__iter_traits::difference_type](#) **difference_type**
- typedef [allocator_traits](#)< [_Alloc](#) >::size_type **size_type**
- typedef [_Alloc](#) **allocator_type**
- typedef [__iter_traits::value_type](#) **char_type**
- typedef [std::basic_string](#)< [char_type](#) > **string_type**

28.10.1 Construction, Copying, and Destruction

- [match_results](#) (const [_Alloc](#) &__a=[_Alloc](#)())
- [match_results](#) (const [match_results](#) &__rhs)=default
- [match_results](#) ([match_results](#) &&__rhs) noexcept=default
- [match_results](#) & [operator=](#) (const [match_results](#) &__rhs)=default
- [match_results](#) & [operator=](#) ([match_results](#) &&__rhs)=default
- [~match_results](#) ()

28.10.2 Size

- size_type [size](#) () const
- size_type [max_size](#) () const
- bool [empty](#) () const

10.3 Element Access

- difference_type [length](#) (size_type __sub=0) const
- difference_type [position](#) (size_type __sub=0) const
- [string_type](#) [str](#) (size_type __sub=0) const
- [const_reference](#) [operator\[\]](#) (size_type __sub) const
- [const_reference](#) [prefix](#) () const
- [const_reference](#) [suffix](#) () const
- const_iterator [begin](#) () const
- const_iterator [cbegin](#) () const
- const_iterator [end](#) () const
- const_iterator [cend](#) () const

10.4 Formatting

These functions perform formatted substitution of the matched character sequences into their target. The format specifiers and escape sequences accepted by these functions are determined by their `flags` parameter as documented above.

- `template<typename _Out_iter>`
`_Out_iter format (_Out_iter __out, const char_type * __fmt_first, const char_type * __fmt_last, match_flag_type`
`__flags=regex_constants::format_default) const`
- `template<typename _Out_iter, typename _St, typename _Sa>`
`_Out_iter format (_Out_iter __out, const basic_string< char_type, _St, _Sa > & __fmt, match_flag_type __`
`flags=regex_constants::format_default) const`
- `template<typename _St, typename _Sa>`
`basic_string< char_type, _St, _Sa > format (const basic_string< char_type, _St, _Sa > & __fmt, match_flag_`
`type __flags=regex_constants::format_default) const`
- `string_type format (const char_type * __fmt, match_flag_type __flags=regex_constants::format_default) const`

10.5 Allocator

- `allocator_type get_allocator () const`

10.6 Swap

- `void swap (match_results & __that)`

5.863.1 Detailed Description

```
template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter>>>
class std::match_results<_Bi_iter, _Alloc>
```

The results of a match or search operation.

A collection of character sequences representing the result of a regular expression match. Storage for the collection is allocated and freed as necessary by the member functions of class template `match_results`.

This class satisfies the Sequence requirements, with the exception that only the operations defined for a const-qualified Sequence are supported.

The `sub_match` object stored at index 0 represents sub-expression 0, i.e. the whole match. In this case the `sub_`↵
`match` member `matched` is always true. The `sub_match` object stored at index `n` denotes what matched the marked
sub-expression `n` within the matched expression. If the sub-expression `n` participated in a regular expression match
then the `sub_match` member `matched` evaluates to true, and members `first` and `second` denote the range of characters
[`first`, `second`) which formed that match. Otherwise `matched` is false, and members `first` and `second` point to the end of
the sequence that was searched.

Definition at line 39 of file `regex.h`.

5.863.2 Constructor & Destructor Documentation

5.863.2.1 `template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter>>> std::match_results<_Bi_iter, _Alloc>::match_results (const _Alloc & __a = _Alloc()) [inline], [explicit]`

Constructs a default match_results container.

Postcondition

size() returns 0 and str() returns an empty string.

Definition at line 1565 of file regex.h.

5.863.2.2 `template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter>>> std::match_results<_Bi_iter, _Alloc>::match_results (const match_results<_Bi_iter, _Alloc> & __rhs) [default]`

Copy constructs a match_results.

5.863.2.3 `template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter>>> std::match_results<_Bi_iter, _Alloc>::match_results (match_results<_Bi_iter, _Alloc> && __rhs) [default], [noexcept]`

Move constructs a match_results.

5.863.2.4 `template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter>>> std::match_results<_Bi_iter, _Alloc>::~~match_results () [inline]`

Destroys a match_results object.

Definition at line 1594 of file regex.h.

5.863.3 Member Function Documentation

5.863.3.1 `template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter>>> const_iterator std::match_results<_Bi_iter, _Alloc>::begin () const [inline]`

Gets an iterator to the start of the sub_match collection.

Definition at line 1739 of file regex.h.

Referenced by std::operator==().

5.863.3.2 `template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter>>> const_iterator std::match_results<_Bi_iter, _Alloc>::cbegin () const [inline]`

Gets an iterator to the start of the sub_match collection.

Definition at line 1746 of file regex.h.

5.863.3.3 `template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter>>> const_iterator std::match_results<_Bi_iter, _Alloc>::cend () const [inline]`

Gets an iterator to one-past-the-end of the collection.

Definition at line 1760 of file regex.h.

5.863.3.4 `template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter>>> bool std::match_results<_Bi_iter, _Alloc>::empty () const [inline]`

Indicates if the match_results contains no results.

Return values

<i>true</i>	The match_results object is empty.
<i>false</i>	The match_results object is not empty.

Definition at line 1635 of file regex.h.

Referenced by std::operator==(), and std::regex_iterator<_Bi_iter, _Ch_type, _Rx_traits >::operator==().

5.863.3.5 `template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter> >> const_iterator
std::match_results<_Bi_iter, _Alloc>::end () const [inline]`

Gets an iterator to one-past-the-end of the collection.

Definition at line 1753 of file regex.h.

Referenced by std::operator==().

5.863.3.6 `template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter> >> template<typename _Out_iter >
_Out_iter std::match_results<_Bi_iter, _Alloc>::format (_Out_iter __out, const char_type * __fmt_first, const
char_type * __fmt_last, match_flag_type __flags = regex_constants::format_default) const`

Precondition

`ready() == true`

Referenced by std::regex_traits<_Ch_type >::value().

5.863.3.7 `template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter> >> template<typename _Out_iter ,
typename _St , typename _Sa > _Out_iter std::match_results<_Bi_iter, _Alloc>::format (_Out_iter __out, const
basic_string<char_type, _St, _Sa> & __fmt, match_flag_type __flags = regex_constants::format_default)
const [inline]`

Precondition

`ready() == true`

Definition at line 1789 of file regex.h.

5.863.3.8 `template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter> >> template<typename _St ,
typename _Sa > basic_string<char_type, _St, _Sa> std::match_results<_Bi_iter, _Alloc>::format (const
basic_string<char_type, _St, _Sa> & __fmt, match_flag_type __flags = regex_constants::format_default)
const [inline]`

Precondition

`ready() == true`

Definition at line 1801 of file regex.h.


```
5.863.3.9  template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter> >> string_type
std::match_results<_Bi_iter, _Alloc>::format ( const char_type * __fmt, match_flag_type __flags =
regex_constants::format_default ) const  [inline]
```

Precondition

ready() == true

Definition at line 1813 of file regex.h.

```
5.863.3.10 template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter> >> allocator_type
std::match_results<_Bi_iter, _Alloc>::get_allocator ( ) const  [inline]
```

Gets a copy of the allocator.

Definition at line 1835 of file regex.h.

```
5.863.3.11 template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter> >> difference_type
std::match_results<_Bi_iter, _Alloc>::length ( size_type __sub = 0 ) const  [inline]
```

Gets the length of the indicated submatch.

Parameters

<code>__sub</code>	indicates the submatch.
--------------------	-------------------------

Precondition

ready() == true

This function returns the length of the indicated submatch, or the length of the entire match if `__sub` is zero (the default).

Definition at line 1654 of file regex.h.

```
5.863.3.12 template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter> >> size_type
std::match_results<_Bi_iter, _Alloc>::max_size ( ) const  [inline]
```

Gets the number of matches and submatches.

The number of matches for a given regular expression will be either 0 if there was no match or `mark_count() + 1` if a match was successful. Some matches may be empty.

Returns

the number of matches found.

Definition at line 1626 of file regex.h.

```
5.863.3.13 template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter> >> match_results&
std::match_results<_Bi_iter, _Alloc >::operator= ( const match_results<_Bi_iter, _Alloc > &__rhs )
[default]
```

Assigns rhs to *this.

```
5.863.3.14 template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter> >> match_results&
std::match_results<_Bi_iter, _Alloc >::operator= ( match_results<_Bi_iter, _Alloc > &&__rhs )
[default]
```

Move-assigns rhs to *this.

```
5.863.3.15 template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter> >> const_reference
std::match_results<_Bi_iter, _Alloc >::operator[]( size_type __sub ) const [inline]
```

Gets a sub_match reference for the match or submatch.

Parameters

<code>__sub</code>	indicates the submatch.
--------------------	-------------------------

Precondition

`ready() == true`

This function gets a reference to the indicated submatch, or the entire match if `__sub` is zero.

If `__sub >= size()` then this function returns a sub_match with a special value indicating no submatch.

Definition at line 1697 of file regex.h.

```
5.863.3.16 template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter> >> difference_type
std::match_results<_Bi_iter, _Alloc >::position ( size_type __sub = 0 ) const [inline]
```

Gets the offset of the beginning of the indicated submatch.

Parameters

<code>__sub</code>	indicates the submatch.
--------------------	-------------------------

Precondition

`ready() == true`

This function returns the offset from the beginning of the target sequence to the beginning of the submatch, unless the value of `__sub` is zero (the default), in which case this function returns the offset from the beginning of the target sequence to the beginning of the match.

Definition at line 1669 of file regex.h.

5.863.3.17 `template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter>>> const_reference
std::match_results<_Bi_iter, _Alloc>::prefix () const [inline]`

Gets a sub_match representing the match prefix.

Precondition

`ready() == true`

This function gets a reference to a sub_match object representing the part of the target range between the start of the target range and the start of the match.

Definition at line 1714 of file regex.h.

Referenced by `std::operator==()`.

5.863.3.18 `template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter>>> bool std::match_results<
_Bi_iter, _Alloc>::ready () const [inline]`

Indicates if the match_results is ready.

Return values

<i>true</i>	The object has a fully-established result state.
<i>false</i>	The object is not ready.

Definition at line 1605 of file regex.h.

Referenced by `std::operator==()`.

5.863.3.19 `template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter>>> size_type
std::match_results<_Bi_iter, _Alloc>::size () const [inline]`

Gets the number of matches and submatches.

The number of matches for a given regular expression will be either 0 if there was no match or `mark_count() + 1` if a match was successful. Some matches may be empty.

Returns

the number of matches found.

Definition at line 1622 of file regex.h.

Referenced by `std::operator==()`.

5.863.3.20 `template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter>>> string_type
std::match_results<_Bi_iter, _Alloc>::str (size_type __sub = 0) const [inline]`

Gets the match or submatch converted to a string type.

Parameters

<code>__sub</code>	indicates the submatch.
--------------------	-------------------------

Precondition

`ready() == true`

This function gets the submatch (or match, if `__sub` is zero) extracted from the target range and converted to the associated string type.

Definition at line 1682 of file `regex.h`.

```
5.863.3.21  template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter>>> const_reference
            std::match_results<_Bi_iter, _Alloc>::suffix ( ) const    [inline]
```

Gets a `sub_match` representing the match suffix.

Precondition

`ready() == true`

This function gets a reference to a `sub_match` object representing the part of the target range between the end of the match and the end of the target range.

Definition at line 1729 of file `regex.h`.

Referenced by `std::operator==()`.

```
5.863.3.22  template<typename _Bi_iter, typename _Alloc = allocator<sub_match<_Bi_iter>>> void std::match_results<
            _Bi_iter, _Alloc>::swap ( match_results<_Bi_iter, _Alloc> &__that )    [inline]
```

Swaps the contents of two `match_results`.

Definition at line 1849 of file `regex.h`.

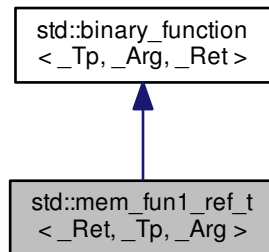
Referenced by `std::swap()`.

The documentation for this class was generated from the following files:

- [regex.h](#)
- [regex.tcc](#)

5.864 `std::mem_fun1_ref_t<_Ret, _Tp, _Arg>` Class Template Reference

Inheritance diagram for `std::mem_fun1_ref_t<_Ret, _Tp, _Arg>`:



Public Types

- typedef `_Tp` [first_argument_type](#)
- typedef `_Ret` [result_type](#)
- typedef `_Arg` [second_argument_type](#)

Public Member Functions

- **`mem_fun1_ref_t`** (`_Ret` (`_Tp::*__pf`) (`_Arg`))
- `_Ret` **`operator()`** (`_Tp &__r`, `_Arg __x`) const

5.864.1 Detailed Description

```
template<typename _Ret, typename _Tp, typename _Arg>
class std::mem_fun1_ref_t<_Ret, _Tp, _Arg>
```

One of the [adaptors for member pointers](#).

Definition at line 1046 of file `stl_function.h`.

5.864.2 Member Typedef Documentation

5.864.2.1 typedef `_Tp` `std::binary_function<_Tp, _Arg, _Ret>::first_argument_type` [\[inherited\]](#)

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.864.2.2 `typedef _Ret std::binary_function<_Tp, _Arg, _Ret>::result_type` [inherited]

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.864.2.3 `typedef _Arg std::binary_function<_Tp, _Arg, _Ret>::second_argument_type` [inherited]

`second_argument_type` is the type of the second argument

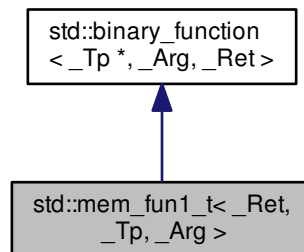
Definition at line 124 of file `stl_function.h`.

The documentation for this class was generated from the following file:

- [stl_function.h](#)

5.865 `std::mem_fun1_t<_Ret, _Tp, _Arg>` Class Template Reference

Inheritance diagram for `std::mem_fun1_t<_Ret, _Tp, _Arg>`:



Public Types

- `typedef _Tp *` [first_argument_type](#)
- `typedef _Ret` [result_type](#)
- `typedef _Arg` [second_argument_type](#)

Public Member Functions

- `mem_fun1_t` (`_Ret` (`_Tp::*` `__pf`) (`_Arg`))
- `_Ret operator()` (`_Tp *` `__p`, `_Arg` `__x`) `const`

5.865.1 Detailed Description

```
template<typename _Ret, typename _Tp, typename _Arg>
class std::mem_fun1_t< _Ret, _Tp, _Arg >
```

One of the [adaptors for member pointers](#).

Definition at line 1010 of file `stl_function.h`.

5.865.2 Member Typedef Documentation

5.865.2.1 `typedef _Tp * std::binary_function< _Tp *, _Arg, _Ret >::first_argument_type` [inherited]

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.865.2.2 `typedef _Ret std::binary_function< _Tp *, _Arg, _Ret >::result_type` [inherited]

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.865.2.3 `typedef _Arg std::binary_function< _Tp *, _Arg, _Ret >::second_argument_type` [inherited]

`second_argument_type` is the type of the second argument

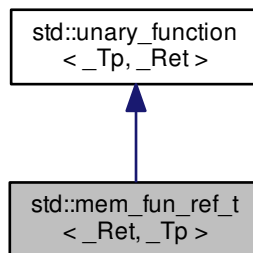
Definition at line 124 of file `stl_function.h`.

The documentation for this class was generated from the following file:

- [stl_function.h](#)

5.866 `std::mem_fun_ref_t< _Ret, _Tp >` Class Template Reference

Inheritance diagram for `std::mem_fun_ref_t< _Ret, _Tp >`:



Public Types

- typedef _Tp [argument_type](#)
- typedef _Ret [result_type](#)

Public Member Functions

- **mem_fun_ref_t** (_Ret(_Tp::*__pf)())
- _Ret **operator()** (_Tp &__r) const

5.866.1 Detailed Description

```
template<typename _Ret, typename _Tp>  
class std::mem_fun_ref_t<_Ret, _Tp>
```

One of the [adaptors for member pointers](#).

Definition at line 974 of file `stl_function.h`.

5.866.2 Member Typedef Documentation

5.866.2.1 typedef _Tp std::unary_function<_Tp, _Ret>::[argument_type](#) [inherited]

`argument_type` is the type of the argument

Definition at line 108 of file `stl_function.h`.

5.866.2.2 typedef _Ret std::unary_function<_Tp, _Ret>::[result_type](#) [inherited]

`result_type` is the return type

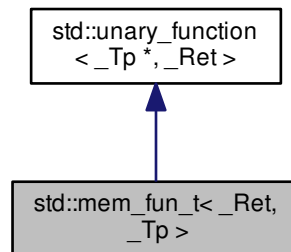
Definition at line 111 of file `stl_function.h`.

The documentation for this class was generated from the following file:

- [stl_function.h](#)

5.867 `std::mem_fun_t<_Ret, _Tp>` Class Template Reference

Inheritance diagram for `std::mem_fun_t<_Ret, _Tp>`:



Public Types

- typedef `_Tp *` [argument_type](#)
- typedef `_Ret` [result_type](#)

Public Member Functions

- `mem_fun_t(_Ret(_Tp::*__pf)())`
- `_Ret operator()(_Tp *__p) const`

5.867.1 Detailed Description

```
template<typename _Ret, typename _Tp>
class std::mem_fun_t<_Ret, _Tp>
```

One of the [adaptors for member pointers](#).

Definition at line 938 of file `stl_function.h`.

5.867.2 Member Typedef Documentation

5.867.2.1 `typedef _Tp * std::unary_function<_Tp*, _Ret>::argument_type` [inherited]

`argument_type` is the type of the argument

Definition at line 108 of file `stl_function.h`.

5.867.2.2 `typedef _Ret std::unary_function<_Tp*, _Ret>::result_type` [inherited]

`result_type` is the return type

Definition at line 111 of file `stl_function.h`.

The documentation for this class was generated from the following file:

- [stl_function.h](#)

5.868 `std::mersenne_twister_engine<_UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f>` **Class Template Reference**

Public Types

- `typedef _UIntType` [result_type](#)

Public Member Functions

- `mersenne_twister_engine` ([result_type](#) __sd=default_seed)
- `template<typename _Sseq, typename = typename std::enable_if<!std::is_same<_Sseq, mersenne_twister_engine>::value>::type>`
`mersenne_twister_engine` (_Sseq &__q)
- `void` [discard](#) (unsigned long long __z)
- `result_type` [operator\(\)](#) ()
- `void` [seed](#) ([result_type](#) __sd=default_seed)
- `template<typename _Sseq>`
`std::enable_if< std::is_class<_Sseq>::value>::type` [seed](#) (_Sseq &__q)

Static Public Member Functions

- `static constexpr` [result_type](#) [max](#) ()
- `static constexpr` [result_type](#) [min](#) ()

Static Public Attributes

- `static constexpr` [result_type](#) [default_seed](#)
- `static constexpr` [result_type](#) [initialization_multiplier](#)
- `static constexpr` `size_t` [mask_bits](#)
- `static constexpr` `size_t` [shift_size](#)
- `static constexpr` `size_t` [state_size](#)
- `static constexpr` [result_type](#) [tempering_b](#)
- `static constexpr` [result_type](#) [tempering_c](#)
- `static constexpr` [result_type](#) [tempering_d](#)
- `static constexpr` `size_t` [tempering_l](#)
- `static constexpr` `size_t` [tempering_s](#)
- `static constexpr` `size_t` [tempering_t](#)
- `static constexpr` `size_t` [tempering_u](#)
- `static constexpr` `size_t` [word_size](#)
- `static constexpr` [result_type](#) [xor_mask](#)

Friends

- `template<typename _UIntType1, size_t __w1, size_t __n1, size_t __m1, size_t __r1, _UIntType1 __a1, size_t __u1, _UIntType1 __d1, size_t __s1, _UIntType1 __b1, size_t __t1, _UIntType1 __c1, size_t __l1, _UIntType1 __f1, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > & __os, const`
`std::mersenne_twister_engine< _UIntType1, __w1, __n1, __m1, __r1, __a1, __u1, __d1, __s1, __b1, __t1, _↵`
`__c1, __l1, __f1 > & __x)`
- `bool operator== (const mersenne_twister_engine & __lhs, const mersenne_twister_engine & __rhs)`
- `template<typename _UIntType1, size_t __w1, size_t __n1, size_t __m1, size_t __r1, _UIntType1 __a1, size_t __u1, _UIntType1 __d1, size_t __s1, _UIntType1 __b1, size_t __t1, _UIntType1 __c1, size_t __l1, _UIntType1 __f1, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > & __is, std↵`
`::mersenne_twister_engine< _UIntType1, __w1, __n1, __m1, __r1, __a1, __u1, __d1, __s1, __b1, __t1, __c1,`
`__l1, __f1 > & __x)`

5.868.1 Detailed Description

```
template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s,
_UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f>
class std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >
```

A generalized feedback shift register discrete random number generator.

This algorithm avoids multiplication and division and is designed to be friendly to a pipelined architecture. If the parameters are chosen correctly, this generator will produce numbers with a very long period and fairly good apparent entropy, although still not cryptographically strong.

The best way to use this generator is with the predefined `mt19937` class.

This algorithm was originally invented by Makoto Matsumoto and Takuji Nishimura.

Template Parameters

<code>_↵ __w</code>	Word size, the number of bits in each element of the state vector.
<code>_↵ __n</code>	The degree of recursion.
<code>_↵ __m</code>	The period parameter.
<code>_↵ __r</code>	The separation point bit index.
<code>_↵ __a</code>	The last row of the twist matrix.
<code>_↵ __u</code>	The first right-shift tempering matrix parameter.
<code>_↵ __d</code>	The first right-shift tempering matrix mask.
<code>_↵ __s</code>	The first left-shift tempering matrix parameter.
<code>_↵ __b</code>	The first left-shift tempering matrix mask.

Template Parameters

<div><div>↔</div><div><code>__t</code></div></div>	The second left-shift tempering matrix parameter.
<div><div>↔</div><div><code>__c</code></div></div>	The second left-shift tempering matrix mask.
<div><div>↔</div><div><code>__l</code></div></div>	The second right-shift tempering matrix parameter.
<div><div>↔</div><div><code>__f</code></div></div>	Initialization multiplier.

Definition at line 444 of file random.h.

5.868.2 Member Typedef Documentation

5.868.2.1 `template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f> typedef _UIntType std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >::result_type`

The type of the generated random value.

Definition at line 447 of file random.h.

5.868.3 Constructor & Destructor Documentation

5.868.3.1 `template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f> template<typename _Sseq, typename = typename std::enable_if<!std::is_same< _Sseq, mersenne_twister_engine>::value> ::type> std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >::mersenne_twister_engine (_Sseq & __q) [inline], [explicit]`

Constructs a mersenne_twister_engine random number generator engine seeded from the seed sequence __q.

Parameters

<div><div>↔</div><div><code>__q</code></div></div>	the seed sequence.
--	--------------------

Definition at line 508 of file random.h.

5.868.4 Member Function Documentation

5.868.4.1 `template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f> void std::mersenne_twister_engine<_UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f>::discard (unsigned long long __z)`

Discard a sequence of random numbers.

Definition at line 435 of file bits/random.tcc.

References `std::dec()`, `std::fixed()`, `std::ios_base::flags()`, `std::left()`, `std::__detail::operator>>()`, `std::subtract_with_carry_engine<_UIntType, __w, __s, __r>::seed()`, and `std::skipws()`.

5.868.4.2 `template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f> static constexpr result_type std::mersenne_twister_engine<_UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f>::max () [inline], [static]`

Gets the largest possible value in the output range.

Definition at line 529 of file random.h.

5.868.4.3 `template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f> static constexpr result_type std::mersenne_twister_engine<_UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f>::min () [inline], [static]`

Gets the smallest possible value in the output range.

Definition at line 522 of file random.h.

5.868.5 Friends And Related Function Documentation

5.868.5.1 `template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f> template<typename _UIntType1, size_t __w1, size_t __n1, size_t __m1, size_t __r1, _UIntType1 __a1, size_t __u1, _UIntType1 __d1, size_t __s1, _UIntType1 __b1, size_t __t1, _UIntType1 __c1, size_t __l1, _UIntType1 __f1, typename _CharT, typename _Traits> std::basic_ostream<_CharT, _Traits>& operator<< (std::basic_ostream<_CharT, _Traits> & __os, const std::mersenne_twister_engine<_UIntType1, __w1, __n1, __m1, __r1, __a1, __u1, __d1, __s1, __b1, __t1, __c1, __l1, __f1> & __x) [friend]`

Inserts the current state of a % mersenne_twister_engine random number generator engine __x into the output stream __os.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A % mersenne_twister_engine random number generator engine.

Returns

The output stream with the state of `__x` inserted or in an error state.

5.868.5.2 `template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f> bool operator==(const mersenne_twister_engine<_UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f> &__lhs, const mersenne_twister_engine<_UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f> &__rhs) [friend]`

Compares two % mersenne_twister_engine random number generator objects of the same type for equality.

Parameters

<code>__lhs</code>	A % mersenne_twister_engine random number generator object.
<code>__rhs</code>	Another % mersenne_twister_engine random number generator object.

Returns

true if the infinite sequences of generated values would be equal, false otherwise.

Definition at line 554 of file random.h.

5.868.5.3 `template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f> template<typename _UIntType1, size_t __w1, size_t __n1, size_t __m1, size_t __r1, _UIntType1 __a1, size_t __u1, _UIntType1 __d1, size_t __s1, _UIntType1 __b1, size_t __t1, _UIntType1 __c1, size_t __l1, _UIntType1 __f1, typename _CharT, typename _Traits> std::basic_istream<_CharT, _Traits>& operator>> (std::basic_istream<_CharT, _Traits> &__is, std::mersenne_twister_engine<_UIntType1, __w1, __n1, __m1, __r1, __a1, __u1, __d1, __s1, __b1, __t1, __c1, __l1, __f1> &__x) [friend]`

Extracts the current state of a % mersenne_twister_engine random number generator engine `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A % mersenne_twister_engine random number generator engine.

Returns

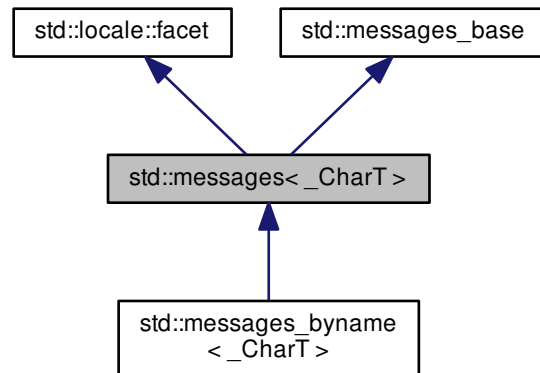
The input stream with the state of `__x` extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.869 `std::messages<_CharT>` Class Template Reference

Inheritance diagram for `std::messages<_CharT>`:



Public Types

- typedef int **catalog**
- typedef `_CharT` **char_type**
- typedef `basic_string<_CharT>` **string_type**

Public Member Functions

- **messages** (size_t __refs=0)
- **messages** (__c_locale __cloc, const char *__s, size_t __refs=0)
- void **close** (catalog __c) const
- **string_type** **get** (catalog __c, int __set, int __msgid, const **string_type** &__s) const
- catalog **open** (const **basic_string**< char > &__s, const **locale** &__loc) const
- catalog **open** (const **basic_string**< char > &, const **locale** &, const char *) const

Static Public Attributes

- static **locale::id** **id**

Protected Member Functions

- virtual `~messages()`
- `string_type _M_convert_from_char(char*) const`
- `char* _M_convert_to_char(const string_type &__msg) const`
- `template<> void do_close(catalog) const`
- `template<> void do_close(catalog) const`
- virtual `void do_close(catalog) const`
- virtual `string_type do_get(catalog, int, int, const string_type &__default) const`
- `template<> string do_get(catalog, int, int, const string &) const`
- `template<> wstring do_get(catalog, int, int, const wstring &) const`
- `template<> messages<char>::catalog do_open(const basic_string<char> &, const locale &) const`
- `template<> messages<wchar_t>::catalog do_open(const basic_string<char> &, const locale &) const`
- virtual `catalog do_open(const basic_string<char> &, const locale &) const`

Static Protected Member Functions

- static `__c_locale _S_clone_c_locale(__c_locale &__cloc) throw()`
- static `void _S_create_c_locale(__c_locale &__cloc, const char* __s, __c_locale __old=0)`
- static `void _S_destroy_c_locale(__c_locale &__cloc)`
- static `__c_locale _S_get_c_locale()`
- static `const char* _S_get_c_name() throw()`
- static `__c_locale _S_lc_ctype_c_locale(__c_locale __cloc, const char* __s)`

Protected Attributes

- `__c_locale _M_c_locale_messages`
- `const char* _M_name_messages`

5.869.1 Detailed Description

```
template<typename _CharT>
class std::messages<_CharT>
```

Primary class template `messages`.

This facet encapsulates the code to retrieve messages from message catalogs. The only thing defined by the standard for this facet is the interface. All underlying functionality is implementation-defined.

This library currently implements 3 versions of the message facet. The first version (gnu) is a wrapper around `gettext`, provided by `libintl`. The second version (ieee) is a wrapper around `catgets`. The final version (default) does no actual translation. These implementations are only provided for `char` and `wchar_t` instantiations.

The `messages` template uses protected virtual functions to provide the actual results. The public accessors forward the call to the virtual functions. These virtual functions are hooks for developers to implement the behavior they require from the `messages` facet.

Definition at line 1797 of file `locale_facets_nonio.h`.

5.869.2 Member Typedef Documentation

5.869.2.1 `template<typename _CharT> typedef _CharT std::messages<_CharT>::char_type`

Public typedefs.

Definition at line 1803 of file locale_facets_nonio.h.

5.869.2.2 `template<typename _CharT> typedef basic_string<_CharT> std::messages<_CharT>::string_type`

Public typedefs.

Definition at line 1804 of file locale_facets_nonio.h.

5.869.3 Constructor & Destructor Documentation

5.869.3.1 `template<typename _CharT> std::messages<_CharT>::messages (size_t __refs = 0) [explicit]`

Constructor performs initialization.

This is the constructor provided by the standard.

Parameters

<code>__refs</code>	Passed to the base facet class.
---------------------	---------------------------------

Definition at line 44 of file messages_members.h.

5.869.3.2 `template<typename _CharT> std::messages<_CharT>::messages (__c_locale __cloc, const char * __s, size_t __refs = 0) [explicit]`

Internal constructor. Not for general use.

This is a constructor for use by the library itself to set up new locales.

Parameters

<code>__cloc</code>	The C locale.
<code>__s</code>	The name of a locale.
<code>__refs</code>	Refcount to pass to the base class.

Definition at line 50 of file messages_members.h.

References `std::basic_string<_CharT, _Traits, _Alloc>::c_str()`.

5.869.3.3 `template<typename _CharT> std::messages<_CharT>::~~messages ()` `[protected]`, `[virtual]`

Destructor.

Definition at line 79 of file `messages_members.h`.

References `std::basic_string<_CharT, _Traits, _Alloc>::c_str()`.

5.869.4 Member Function Documentation

5.869.4.1 `template<> string std::messages<char>::do_get (catalog , int , int , const string &) const`
`[protected]`

Specializations for required instantiations.

5.869.5 Member Data Documentation

5.869.5.1 `template<typename _CharT> locale::id std::messages<_CharT>::id` `[static]`

Numpunct facet id.

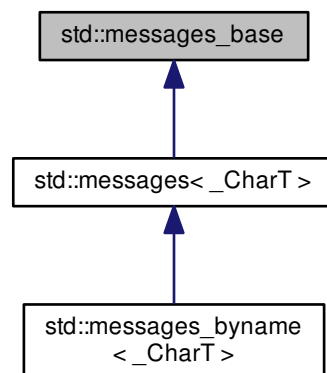
Definition at line 1815 of file `locale_facets_nonio.h`.

The documentation for this class was generated from the following files:

- [locale_facets_nonio.h](#)
- [messages_members.h](#)

5.870 std::messages_base Struct Reference

Inheritance diagram for `std::messages_base`:



Public Types

- typedef int **catalog**

5.870.1 Detailed Description

Messages facet base class providing catalog typedef.

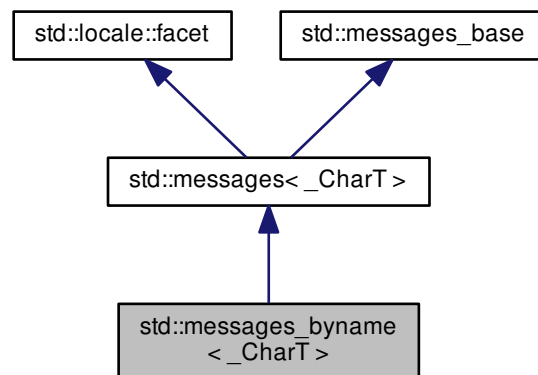
Definition at line 1768 of file locale_facets_nonio.h.

The documentation for this struct was generated from the following file:

- [locale_facets_nonio.h](#)

5.871 std::messages_byname<_CharT> Class Template Reference

Inheritance diagram for std::messages_byname<_CharT>:



Public Types

- typedef int **catalog**
- typedef `_CharT` **char_type**
- typedef [basic_string<_CharT>](#) **string_type**

Public Member Functions

- **messages_byname** (const char *__s, size_t __refs=0)
- **messages_byname** (const [string](#) &__s, size_t __refs=0)
- void **close** (catalog __c) const
- [string_type](#) **get** (catalog __c, int __set, int __msgid, const [string_type](#) &__s) const
- catalog **open** (const [basic_string](#)< char > &__s, const [locale](#) &__loc) const
- catalog **open** (const [basic_string](#)< char > &, const [locale](#) &, const char *) const

Static Public Attributes

- static [locale::id](#) id

Protected Member Functions

- [string_type](#) **_M_convert_from_char** (char *) const
- char * **_M_convert_to_char** (const [string_type](#) &__msg) const
- template<>
void **do_close** (catalog) const
- template<>
void **do_close** (catalog) const
- virtual void **do_close** (catalog) const
- virtual [string_type](#) **do_get** (catalog, int, int, const [string_type](#) &__dfault) const
- template<>
[string](#) **do_get** (catalog, int, int, const [string](#) &) const
- template<>
[wstring](#) **do_get** (catalog, int, int, const [wstring](#) &) const
- template<>
[messages](#)< char >::catalog **do_open** (const [basic_string](#)< char > &, const [locale](#) &) const
- template<>
[messages](#)< wchar_t >::catalog **do_open** (const [basic_string](#)< char > &, const [locale](#) &) const
- virtual catalog **do_open** (const [basic_string](#)< char > &, const [locale](#) &) const

Static Protected Member Functions

- static __c_locale **_S_clone_c_locale** (__c_locale &__cloc) throw ()
- static void **_S_create_c_locale** (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale &__cloc)
- static __c_locale **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static __c_locale **_S_lc_type_c_locale** (__c_locale __cloc, const char *__s)

Protected Attributes

- __c_locale **_M_c_locale_messages**
- const char * **_M_name_messages**

5.871.1 Detailed Description

```
template<typename _CharT>
class std::messages_byname< _CharT >
```

class messages_byname [22.2.7.2].

Definition at line 1981 of file locale_facets_nonio.h.

5.871.2 Member Function Documentation

5.871.2.1 `template<> string std::messages< char >::do_get (catalog , int , int , const string &) const`
`[protected], [inherited]`

Specializations for required instantiations.

5.871.3 Member Data Documentation

5.871.3.1 `template<typename _CharT> locale::id std::messages< _CharT >::id` `[static], [inherited]`

Numpunct facet id.

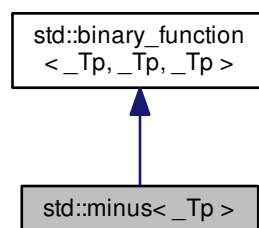
Definition at line 1815 of file locale_facets_nonio.h.

The documentation for this class was generated from the following files:

- [locale_facets_nonio.h](#)
- [messages_members.h](#)

5.872 std::minus< _Tp > Struct Template Reference

Inheritance diagram for std::minus< _Tp >:



Public Types

- typedef _Tp [first_argument_type](#)
- typedef _Tp [result_type](#)
- typedef _Tp [second_argument_type](#)

Public Member Functions

- `_GLIBCXX14_CONSTEXPR _Tp operator() (const _Tp &__x, const _Tp &__y) const`

5.872.1 Detailed Description

```
template<typename _Tp>
struct std::minus<_Tp>
```

One of the [math functors](#).

Definition at line 150 of file `stl_function.h`.

5.872.2 Member Typedef Documentation

5.872.2.1 `typedef _Tp std::binary_function<_Tp, _Tp, _Tp>::first_argument_type` [\[inherited\]](#)

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.872.2.2 `typedef _Tp std::binary_function<_Tp, _Tp, _Tp>::result_type` [\[inherited\]](#)

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.872.2.3 `typedef _Tp std::binary_function<_Tp, _Tp, _Tp>::second_argument_type` [\[inherited\]](#)

`second_argument_type` is the type of the second argument

Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.873 `std::minus< void >` Struct Template Reference

Public Types

- typedef `__is_transparent` is `__transparent`

Public Member Functions

- template<typename `_Tp`, typename `_Up` >
`_GLIBCXX14_CONSTEXPR` auto **operator()** (`_Tp` &&`__t`, `_Up` &&`__u`) const noexcept(noexcept(`std::forward`<
`_Tp` >(`__t`)-`std::forward`< `_Up` >(`__u`))) -> decltype(`std::forward`< `_Tp` >(`__t`)-`std::forward`< `_Up` >(`__u`))

5.873.1 Detailed Description

```
template<>
struct std::minus< void >
```

One of the [math functions](#).

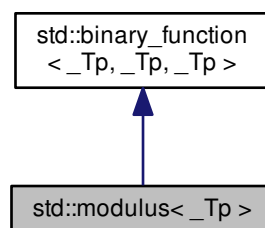
Definition at line 245 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.874 `std::modulus< _Tp >` Struct Template Reference

Inheritance diagram for `std::modulus< _Tp >`:



Public Types

- typedef _Tp [first_argument_type](#)
- typedef _Tp [result_type](#)
- typedef _Tp [second_argument_type](#)

Public Member Functions

- `_GLIBCXX14_CONSTEXPR _Tp operator() (const _Tp &__x, const _Tp &__y) const`

5.874.1 Detailed Description

```
template<typename _Tp>
struct std::modulus<_Tp>
```

One of the [math functors](#).

Definition at line 159 of file `stl_function.h`.

5.874.2 Member Typedef Documentation

5.874.2.1 `typedef _Tp std::binary_function<_Tp, _Tp, _Tp>::first_argument_type` [\[inherited\]](#)

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.874.2.2 `typedef _Tp std::binary_function<_Tp, _Tp, _Tp>::result_type` [\[inherited\]](#)

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.874.2.3 `typedef _Tp std::binary_function<_Tp, _Tp, _Tp>::second_argument_type` [\[inherited\]](#)

`second_argument_type` is the type of the second argument

Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.875 `std::modulus< void >` Struct Template Reference

Public Types

- typedef `__is_transparent` is `__transparent`

Public Member Functions

- template<typename `_Tp` , typename `_Up` >
`_GLIBCXX14_CONSTEXPR` auto **operator()** (`_Tp` &&`__t`, `_Up` &&`__u`) const noexcept(noexcept(`std::forward< _Tp >(__t)%std::forward< _Up >(__u)`)) -> `decltype(std::forward< _Tp >(__t)%std::forward< _Up >(__u))`

5.875.1 Detailed Description

```
template<>
struct std::modulus< void >
```

One of the [math functors](#).

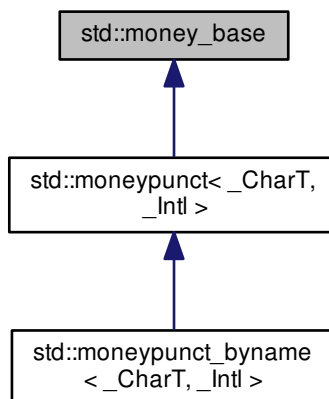
Definition at line 290 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.876 `std::money_base` Class Reference

Inheritance diagram for `std::money_base`:



Public Types

- enum { **_S_minus**, **_S_zero**, **_S_end** }
- enum **part** {
 none, **space**, **symbol**, **sign**,
 value }

Static Public Member Functions

- static pattern **_S_construct_pattern** (char __precedes, char __space, char __posn) throw ()

Static Public Attributes

- static const char * **_S_atoms**
- static const pattern **_S_default_pattern**

5.876.1 Detailed Description

Money format ordering data.

This class contains an ordered array of 4 fields to represent the pattern for formatting a money amount. Each field may contain one entry from the part enum. symbol, sign, and value must be present and the remaining field must contain either none or space.

See also

money_punct::pos_format() and money_punct::neg_format() for details of how these fields are interpreted.

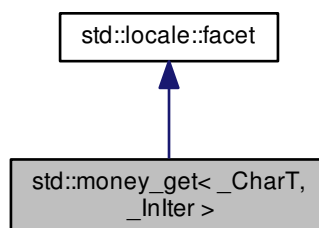
Definition at line 926 of file locale_facets_nonio.h.

The documentation for this class was generated from the following file:

- [locale_facets_nonio.h](#)

5.877 std::money_get< _CharT, _InIter > Class Template Reference

Inheritance diagram for std::money_get< _CharT, _InIter >:



Public Types

- typedef `_CharT` `char_type`
- typedef `_InIter` `iter_type`
- typedef `basic_string<_CharT>` `string_type`

Public Member Functions

- `money_get` (`size_t` __refs=0)
- `template<bool _Intl>`
`_GLIBCXX_BEGIN_NAMESPACE_LDBL_OR_CXX11` `_M_extract` (`iter_type` __beg, `iter_type` __end, `ios_base &__io`, `ios_base::iostate &__err`, `string &__units`) `const`
- `iter_type get` (`iter_type` __s, `iter_type` __end, `bool __intl`, `ios_base &__io`, `ios_base::iostate &__err`, `long double &__units`) `const`
- `iter_type get` (`iter_type` __s, `iter_type` __end, `bool __intl`, `ios_base &__io`, `ios_base::iostate &__err`, `string_type &__digits`) `const`

Static Public Attributes

- static `locale::id` `id`

Protected Member Functions

- virtual `~money_get` ()
- `template<bool _Intl>`
`iter_type` `_M_extract` (`iter_type` __s, `iter_type` __end, `ios_base &__io`, `ios_base::iostate &__err`, `string &__digits`) `const`
- virtual `iter_type do_get` (`iter_type` __s, `iter_type` __end, `bool __intl`, `ios_base &__io`, `ios_base::iostate &__err`, `long double &__units`) `const`
- virtual `iter_type do_get` (`iter_type` __s, `iter_type` __end, `bool __intl`, `ios_base &__io`, `ios_base::iostate &__err`, `string_type &__digits`) `const`

Static Protected Member Functions

- static `__c_locale` `_S_clone_c_locale` (`__c_locale &__cloc`) `throw ()`
- static void `_S_create_c_locale` (`__c_locale &__cloc`, `const char *__s`, `__c_locale __old=0`)
- static void `_S_destroy_c_locale` (`__c_locale &__cloc`)
- static `__c_locale` `_S_get_c_locale` ()
- static `const char *` `_S_get_c_name` () `throw ()`
- static `__c_locale` `_S_lc_ctype_c_locale` (`__c_locale __cloc`, `const char *__s`)

5.877.1 Detailed Description

```
template<typename _CharT, typename _InIter>
class std::money_get< _CharT, _InIter >
```

Primary class template money_get.

This facet encapsulates the code to parse and return a monetary amount from a string.

The money_get template uses protected virtual functions to provide the actual results. The public accessors forward the call to the virtual functions. These virtual functions are hooks for developers to implement the behavior they require from the money_get facet.

Definition at line 1466 of file locale_facets_nonio.h.

5.877.2 Member Typedef Documentation

5.877.2.1 `template<typename _CharT, typename _InIter > typedef _CharT std::money_get< _CharT, _InIter >::char_type`

Public typedefs.

Definition at line 1472 of file locale_facets_nonio.h.

5.877.2.2 `template<typename _CharT, typename _InIter > typedef _InIter std::money_get< _CharT, _InIter >::iter_type`

Public typedefs.

Definition at line 1473 of file locale_facets_nonio.h.

5.877.2.3 `template<typename _CharT, typename _InIter > typedef basic_string<_CharT> std::money_get< _CharT, _InIter >::string_type`

Public typedefs.

Definition at line 1474 of file locale_facets_nonio.h.

5.877.3 Constructor & Destructor Documentation

5.877.3.1 `template<typename _CharT, typename _InIter > std::money_get< _CharT, _InIter >::money_get(size_t __refs = 0) [inline],[explicit]`

Constructor performs initialization.

This is the constructor provided by the standard.

Parameters

<code>__refs</code>	Passed to the base facet class.
---------------------	---------------------------------

Definition at line 1488 of file `locale_facets_nonio.h`.

5.877.3.2 `template<typename _CharT, typename _Inlter > virtual std::money_get< _CharT, _Inlter >::~~money_get ()`
`[inline], [protected], [virtual]`

Destructor.

Definition at line 1556 of file `locale_facets_nonio.h`.

5.877.4 Member Function Documentation

5.877.4.1 `template<typename _CharT, typename _Inlter > _Inlter std::money_get< _CharT, _Inlter >::do_get (iter_type`
`__s, iter_type __end, bool __intl, ios_base & __io, ios_base::iostate & __err, long double & __units) const`
`[protected], [virtual]`

Read and parse a monetary value.

This function reads and parses characters representing a monetary value. This function is a hook for derived classes to change the value returned.

See also

`get()` for details.

Definition at line 370 of file `locale_facets_nonio.tcc`.

References `std::basic_string< _CharT, _Traits, _Alloc >::c_str()`.

5.877.4.2 `template<typename _CharT, typename _Inlter > _Inlter std::money_get< _CharT, _Inlter >::do_get (iter_type`
`__s, iter_type __end, bool __intl, ios_base & __io, ios_base::iostate & __err, string_type & __digits) const`
`[protected], [virtual]`

Read and parse a monetary value.

This function reads and parses characters representing a monetary value. This function is a hook for derived classes to change the value returned.

See also

`get()` for details.

Definition at line 383 of file `locale_facets_nonio.tcc`.

References `std::ios_base::M_getloc()`, `std::ios_base::adjustfield`, `std::basic_string< _CharT, _Traits, _Alloc >::append()`, `std::basic_string< _CharT, _Traits, _Alloc >::assign()`, `std::basic_string< _CharT, _Traits, _Alloc >::data()`, `std::money_put< _CharT, _Outlter >::do_put()`, `std::basic_string< _CharT, _Traits, _Alloc >::erase()`, `std::ios_base::flags()`, `std::basic_string< _CharT, _Traits, _Alloc >::insert()`, `std::ios_base::internal`, `std::ios_base::left`, `std::basic_string< _CharT, _Traits, _Alloc >::reserve()`, `std::basic_string< _CharT, _Traits, _Alloc >::resize()`, `std::__ctype_abstract_base< _CharT >::scan_not()`, `std::ios_base::showbase`, `std::basic_string< _CharT, _Traits, _Alloc >::size()`, `std::__ctype_abstract_base< _CharT >::widen()`, and `std::ios_base::width()`.

```
5.877.4.3 template<typename _CharT, typename _InIter > iter_type std::money_get<_CharT, _InIter >::get ( iter_type
    __s, iter_type __end, bool __intl, ios_base & __io, ios_base::iostate & __err, long double & __units ) const
    [inline]
```

Read and parse a monetary value.

This function reads characters from `__s`, interprets them as a monetary value according to `moneypunct` and `ctype` facets retrieved from `io.getloc()`, and returns the result in `units` as an integral value `moneypunct::frac_digits() * the actual amount`. For example, the string \$10.01 in a US locale would store 1001 in `units`.

Any characters not part of a valid money amount are not consumed.

If a money value cannot be parsed from the input stream, sets `err=(err|io.failbit)`. If the stream is consumed before finishing parsing, sets `err=(err|io.failbit|io.eofbit)`. `units` is unchanged if parsing fails.

This function works by returning the result of `do_get()`.

Parameters

<code>__s</code>	Start of characters to parse.
<code>__end</code>	End of characters to parse.
<code>__intl</code>	Parameter to use <code>_facet<moneypunct<CharT,intl> ></code> .
<code>__io</code>	Source of facets and io state.
<code>__err</code>	Error field to set if parsing fails.
<code>__units</code>	Place to store result of parsing.

Returns

Iterator referencing first character beyond valid money amount.

Definition at line 1518 of file `locale_facets_nonio.h`.

```
5.877.4.4 template<typename _CharT, typename _InIter > iter_type std::money_get<_CharT, _InIter >::get ( iter_type
    __s, iter_type __end, bool __intl, ios_base & __io, ios_base::iostate & __err, string_type & __digits ) const
    [inline]
```

Read and parse a monetary value.

This function reads characters from `__s`, interprets them as a monetary value according to `moneypunct` and `ctype` facets retrieved from `io.getloc()`, and returns the result in `digits`. For example, the string \$10.01 in a US locale would store 1001 in `digits`.

Any characters not part of a valid money amount are not consumed.

If a money value cannot be parsed from the input stream, sets `err=(err|io.failbit)`. If the stream is consumed before finishing parsing, sets `err=(err|io.failbit|io.eofbit)`.

This function works by returning the result of `do_get()`.

Parameters

<code>__s</code>	Start of characters to parse.
<code>__end</code>	End of characters to parse.
<code>__intl</code>	Parameter to use <code>_facet<moneypunct<CharT,intl> ></code> .
<code>__io</code>	Source of facets and io state.
<code>__err</code>	Error field to set if parsing fails.
<code>__digits</code>	Place to store result of parsing.

Returns

Iterator referencing first character beyond valid money amount.

Definition at line 1549 of file `locale_facets_nonio.h`.

5.877.5 Member Data Documentation

5.877.5.1 `template<typename _CharT, typename _Inlter> locale::id std::money_get<_CharT, _Inlter>::id` `[static]`

Numpunct facet id.

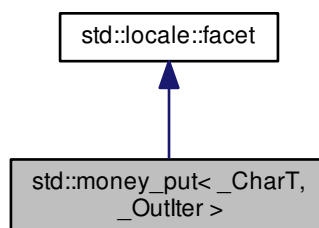
Definition at line 1478 of file `locale_facets_nonio.h`.

The documentation for this class was generated from the following files:

- [locale_facets_nonio.h](#)
- [locale_facets_nonio.tcc](#)

5.878 `std::money_put<_CharT, _Outlter>` Class Template Reference

Inheritance diagram for `std::money_put<_CharT, _Outlter>`:



Public Types

- typedef `_CharT` `char_type`
- typedef `_Outlter` `iter_type`
- typedef `basic_string<_CharT>` `string_type`

Public Member Functions

- `money_put` (`size_t` __refs=0)
- `template<bool __Intl>`
`_Outlter _M_insert` (`iter_type` __s, `ios_base & __io`, `char_type` __fill, `const string_type & __digits`) `const`
- `iter_type put` (`iter_type` __s, `bool __intl`, `ios_base & __io`, `char_type` __fill, `long double __units`) `const`
- `iter_type put` (`iter_type` __s, `bool __intl`, `ios_base & __io`, `char_type` __fill, `const string_type & __digits`) `const`

Static Public Attributes

- static `locale::id` `id`

Protected Member Functions

- virtual `~money_put` ()
- `template<bool __Intl>`
`iter_type _M_insert` (`iter_type` __s, `ios_base & __io`, `char_type` __fill, `const string_type & __digits`) `const`
- virtual `iter_type do_put` (`iter_type` __s, `bool __intl`, `ios_base & __io`, `char_type` __fill, `long double __units`) `const`
- virtual `iter_type do_put` (`iter_type` __s, `bool __intl`, `ios_base & __io`, `char_type` __fill, `const string_type & __digits`) `const`

Static Protected Member Functions

- static `__c_locale _S_clone_c_locale` (`__c_locale & __cloc`) `throw ()`
- static void `_S_create_c_locale` (`__c_locale & __cloc`, `const char * __s`, `__c_locale __old=0`)
- static void `_S_destroy_c_locale` (`__c_locale & __cloc`)
- static `__c_locale _S_get_c_locale` ()
- static `const char * _S_get_c_name` () `throw ()`
- static `__c_locale _S_lc_ctype_c_locale` (`__c_locale __cloc`, `const char * __s`)

5.878.1 Detailed Description

```
template<typename _CharT, typename _Outlter>
class std::money_put<_CharT, _Outlter >
```

Primary class template `money_put`.

This facet encapsulates the code to format and output a monetary amount.

The `money_put` template uses protected virtual functions to provide the actual results. The public accessors forward the call to the virtual functions. These virtual functions are hooks for developers to implement the behavior they require from the `money_put` facet.

Definition at line 1619 of file `locale_facets_nonio.h`.

5.878.2 Member Typedef Documentation

5.878.2.1 `template<typename _CharT, typename _Outiter> typedef _CharT std::money_put<_CharT, _Outiter>::char_type`

Public typedefs.

Definition at line 1624 of file locale_facets_nonio.h.

5.878.2.2 `template<typename _CharT, typename _Outiter> typedef _Outiter std::money_put<_CharT, _Outiter>::iter_type`

Public typedefs.

Definition at line 1625 of file locale_facets_nonio.h.

5.878.2.3 `template<typename _CharT, typename _Outiter> typedef basic_string<_CharT> std::money_put<_CharT, _Outiter>::string_type`

Public typedefs.

Definition at line 1626 of file locale_facets_nonio.h.

5.878.3 Constructor & Destructor Documentation

5.878.3.1 `template<typename _CharT, typename _Outiter> std::money_put<_CharT, _Outiter>::money_put (size_t __refs = 0) [inline], [explicit]`

Constructor performs initialization.

This is the constructor provided by the standard.

Parameters

<code>__refs</code>	Passed to the base facet class.
---------------------	---------------------------------

Definition at line 1640 of file locale_facets_nonio.h.

5.878.3.2 `template<typename _CharT, typename _Outiter> virtual std::money_put<_CharT, _Outiter>::~~money_put () [inline], [protected], [virtual]`

Destructor.

Definition at line 1690 of file locale_facets_nonio.h.

5.878.4 Member Function Documentation

5.878.4.1 `template<typename _CharT, typename _OutIter > _OutIter std::money_put<_CharT, _OutIter>::do_put (iter_type __s, bool __intl, ios_base & __io, char_type __fill, long double __units) const` `[protected]`, `[virtual]`

Format and output a monetary value.

This function formats *units* as a monetary value according to `moneypunct` and `ctype` facets retrieved from `io.getloc()`, and writes the resulting characters to `__s`. For example, the value 1001 in a US locale would write \$10.01 to `__s`.

This function is a hook for derived classes to change the value returned.

See also

`put()`.

Parameters

<code>__s</code>	The stream to write to.
<code>__intl</code>	Parameter to use <code>_facet<moneypunct<CharT,intl>></code> .
<code>__io</code>	Source of facets and io state.
<code>__fill</code>	<code>char_type</code> to use for padding.
<code>__units</code>	Place to store result of parsing.

Returns

Iterator after writing.

Definition at line 576 of file `locale_facets_nonio.tcc`.

References `std::ios_base::getloc()`, and `std::__ctype_abstract_base<_CharT>::widen()`.

Referenced by `std::money_get<_CharT, _InIter>::do_get()`.

5.878.4.2 `template<typename _CharT, typename _OutIter > _OutIter std::money_put<_CharT, _OutIter>::do_put (iter_type __s, bool __intl, ios_base & __io, char_type __fill, const string_type & __digits) const` `[protected]`, `[virtual]`

Format and output a monetary value.

This function formats *digits* as a monetary value according to `moneypunct` and `ctype` facets retrieved from `io.getloc()`, and writes the resulting characters to `__s`. For example, the string 1001 in a US locale would write \$10.01 to `__s`.

This function is a hook for derived classes to change the value returned.

See also

`put()`.

Parameters

<code>__s</code>	The stream to write to.
<code>__intl</code>	Parameter to use <code>_facet<moneypunct<CharT,intl>></code> .
<code>__io</code>	Source of facets and io state.
<code>__fill</code>	<code>char_type</code> to use for padding.
<code>__digits</code>	Place to store result of parsing.

Returns

Iterator after writing.

Definition at line 614 of file `locale_facets_nonio.tcc`.

```
5.878.4.3 template<typename _CharT, typename _Outlter> iter_type std::money_put<_CharT, _Outlter>::put ( iter_type
    __s, bool __intl, ios_base & __io, char_type __fill, long double __units ) const [inline]
```

Format and output a monetary value.

This function formats *units* as a monetary value according to `moneypunct` and `ctype` facets retrieved from `io.getloc()`, and writes the resulting characters to `__s`. For example, the value 1001 in a US locale would write \$10.01 to `__s`.

This function works by returning the result of `do_put()`.

Parameters

<code>__s</code>	The stream to write to.
<code>__intl</code>	Parameter to use <code>_facet<moneypunct<CharT,intl>></code> .
<code>__io</code>	Source of facets and io state.
<code>__fill</code>	<code>char_type</code> to use for padding.
<code>__units</code>	Place to store result of parsing.

Returns

Iterator after writing.

Definition at line 1660 of file `locale_facets_nonio.h`.

```
5.878.4.4 template<typename _CharT, typename _Outlter> iter_type std::money_put<_CharT, _Outlter>::put ( iter_type
    __s, bool __intl, ios_base & __io, char_type __fill, const string_type & __digits ) const [inline]
```

Format and output a monetary value.

This function formats *digits* as a monetary value according to `moneypunct` and `ctype` facets retrieved from `io.getloc()`, and writes the resulting characters to `__s`. For example, the string 1001 in a US locale would write \$10.01 to `__s`.

This function works by returning the result of `do_put()`.

Parameters

<code>__s</code>	The stream to write to.
<code>__intl</code>	Parameter to use <code>_facet<moneypunct<CharT,intl>></code> .
<code>__io</code>	Source of facets and io state.
<code>__fill</code>	<code>char_type</code> to use for padding.
<code>__digits</code>	Place to store result of parsing.

Returns

Iterator after writing.

Definition at line 1683 of file `locale_facets_nonio.h`.

5.878.5 Member Data Documentation

5.878.5.1 `template<typename _CharT , typename _Outlter > locale::id std::money_put< _CharT, _Outlter >::id`
`[static]`

Numpunct facet id.

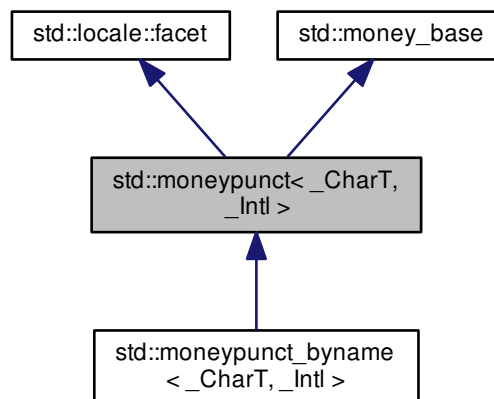
Definition at line 1630 of file `locale_facets_nonio.h`.

The documentation for this class was generated from the following files:

- [locale_facets_nonio.h](#)
- [locale_facets_nonio.tcc](#)

5.879 std::moneypunct< _CharT, _Intl > Class Template Reference

Inheritance diagram for `std::moneypunct< _CharT, _Intl >`:



Public Types

- enum { **_S_minus**, **_S_zero**, **_S_end** }
- typedef __moneypunct_cache< _CharT, _Intl > **__cache_type**
- enum **part** {
 none, **space**, **symbol**, **sign**,
 value }
- typedef _CharT **char_type**
- typedef **basic_string**< _CharT > **string_type**

Public Member Functions

- **moneypunct** (size_t __refs=0)
- **moneypunct** (__cache_type * __cache, size_t __refs=0)
- **moneypunct** (__c_locale __cloc, const char * __s, size_t __refs=0)
- **string_type curr_symbol** () const
- **char_type decimal_point** () const
- int **frac_digits** () const
- **string grouping** () const
- **string_type negative_sign** () const
- **string_type positive_sign** () const
- **char_type thousands_sep** () const
- pattern **pos_format** () const
- pattern **neg_format** () const

Static Public Member Functions

- static pattern **_S_construct_pattern** (char __precedes, char __space, char __posn) throw ()

Static Public Attributes

- static const char * **_S_atoms**
- static const pattern **_S_default_pattern**
- static **locale::id** **id**
- static const bool **intl**

Protected Member Functions

- virtual [~moneypunct](#) ()
- void [_M_initialize_moneypunct](#) (__c_locale __cloc=0, const char *__name=0)
- template<>
void [_M_initialize_moneypunct](#) (__c_locale, const char *)
- template<>
void [_M_initialize_moneypunct](#) (__c_locale, const char *)
- template<>
void [_M_initialize_moneypunct](#) (__c_locale, const char *)
- template<>
void [_M_initialize_moneypunct](#) (__c_locale, const char *)
- virtual [string_type do_curr_symbol](#) () const
- virtual [char_type do_decimal_point](#) () const
- virtual int [do_frac_digits](#) () const
- virtual [string do_grouping](#) () const
- virtual pattern [do_neg_format](#) () const
- virtual [string_type do_negative_sign](#) () const
- virtual pattern [do_pos_format](#) () const
- virtual [string_type do_positive_sign](#) () const
- virtual [char_type do_thousands_sep](#) () const

Static Protected Member Functions

- static __c_locale [_S_clone_c_locale](#) (__c_locale &__cloc) throw ()
- static void [_S_create_c_locale](#) (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void [_S_destroy_c_locale](#) (__c_locale &__cloc)
- static __c_locale [_S_get_c_locale](#) ()
- static const char * [_S_get_c_name](#) () throw ()
- static __c_locale [_S_lc_ctype_c_locale](#) (__c_locale __cloc, const char *__s)

5.879.1 Detailed Description

```
template<typename _CharT, bool _Intl>
class std::moneypunct<_CharT, _Intl>
```

Primary class template moneypunct.

This facet encapsulates the punctuation, grouping and other formatting features of money amount string representations.

Definition at line 1022 of file locale_facets_nonio.h.

5.879.2 Member Typedef Documentation

5.879.2.1 template<typename _CharT, bool _Intl> typedef _CharT std::moneypunct<_CharT, _Intl>::char_type

Public typedefs.

Definition at line 1028 of file locale_facets_nonio.h.

5.879.2.2 `template<typename _CharT, bool _Intl> typedef basic_string<_CharT> std::moneypunct<_CharT, _Intl>::string_type`

Public typedefs.

Definition at line 1029 of file locale_facets_nonio.h.

5.879.3 Constructor & Destructor Documentation

5.879.3.1 `template<typename _CharT, bool _Intl> std::moneypunct<_CharT, _Intl>::moneypunct (size_t __refs = 0) [inline], [explicit]`

Constructor performs initialization.

This is the constructor provided by the standard.

Parameters

<code>__refs</code>	Passed to the base facet class.
---------------------	---------------------------------

Definition at line 1051 of file locale_facets_nonio.h.

5.879.3.2 `template<typename _CharT, bool _Intl> std::moneypunct<_CharT, _Intl>::moneypunct (__cache_type * __cache, size_t __refs = 0) [inline], [explicit]`

Constructor performs initialization.

This is an internal constructor.

Parameters

<code>__cache</code>	Cache for optimization.
<code>__refs</code>	Passed to the base facet class.

Definition at line 1064 of file locale_facets_nonio.h.

5.879.3.3 `template<typename _CharT, bool _Intl> std::moneypunct<_CharT, _Intl>::moneypunct (__c_locale __cloc, const char * __s, size_t __refs = 0) [inline], [explicit]`

Internal constructor. Not for general use.

This is a constructor for use by the library itself to set up new locales.

Parameters

<code>__cloc</code>	The C locale.
<code>__s</code>	The name of a locale.
<code>__refs</code>	Passed to the base facet class.

Definition at line 1079 of file locale_facets_nonio.h.

```
5.879.3.4 template<typename _CharT, bool _Intl> virtual std::moneypunct<_CharT, _Intl>::~~moneypunct ( )  
        [protected], [virtual]
```

Destructor.

Referenced by std::moneypunct<_CharT, _Intl>::do_neg_format().

5.879.4 Member Function Documentation

```
5.879.4.1 template<typename _CharT, bool _Intl> string_type std::moneypunct<_CharT, _Intl>::curr_symbol ( ) const  
        [inline]
```

Return currency symbol string.

This function returns a string_type to use as a currency symbol. It does so by returning returning moneypunct<char↵_type>::do_curr_symbol().

Returns

string_type representing a currency symbol.

Definition at line 1149 of file locale_facets_nonio.h.

```
5.879.4.2 template<typename _CharT, bool _Intl> char_type std::moneypunct<_CharT, _Intl>::decimal_point ( ) const  
        [inline]
```

Return decimal point character.

This function returns a char_type to use as a decimal point. It does so by returning returning moneypunct<char↵_type>::do_decimal_point().

Returns

char_type representing a decimal point.

Definition at line 1093 of file locale_facets_nonio.h.

```
5.879.4.3 template<typename _CharT, bool _Intl> virtual string_type std::moneypunct<_CharT, _Intl>::do_curr_symbol (  
        ) const [inline], [protected], [virtual]
```

Return currency symbol string.

This function returns a string_type to use as a currency symbol. This function is a hook for derived classes to change the value returned.

See also

curr_symbol() for details.

Returns

string_type representing a currency symbol.

Definition at line 1295 of file locale_facets_nonio.h.


```
5.879.4.4  template<typename _CharT, bool _Intl> virtual char_type std::moneypunct<_CharT, _Intl>::do_decimal_point (
    ) const    [inline], [protected], [virtual]
```

Return decimal point character.

Returns a `char_type` to use as a decimal point. This function is a hook for derived classes to change the value returned.

Returns

char_type representing a decimal point.

Definition at line 1257 of file `locale_facets_nonio.h`.

```
5.879.4.5  template<typename _CharT, bool _Intl> virtual int std::moneypunct<_CharT, _Intl>::do_frac_digits ( ) const
    [inline], [protected], [virtual]
```

Return number of digits in fraction.

This function returns the exact number of digits that make up the fractional part of a money amount. This function is a hook for derived classes to change the value returned.

See also

`frac_digits()` for details.

Returns

Number of digits in amount fraction.

Definition at line 1335 of file `locale_facets_nonio.h`.

```
5.879.4.6  template<typename _CharT, bool _Intl> virtual string std::moneypunct<_CharT, _Intl>::do_grouping ( ) const
    [inline], [protected], [virtual]
```

Return grouping specification.

Returns a string representing groupings for the integer part of a number. This function is a hook for derived classes to change the value returned.

See also

`grouping()` for details.

Returns

String representing grouping specification.

Definition at line 1282 of file `locale_facets_nonio.h`.

5.879.4.7 `template<typename _CharT, bool _Intl> virtual pattern std::moneypunct<_CharT, _Intl>::do_neg_format () const`
`[inline], [protected], [virtual]`

Return pattern for money values.

This function returns a pattern describing the formatting of a negative valued money amount. This function is a hook for derived classes to change the value returned.

See also

`neg_format()` for details.

Returns

Pattern for money values.

Definition at line 1363 of file `locale_facets_nonio.h`.

References `std::moneypunct<_CharT, _Intl>::~~moneypunct()`.

5.879.4.8 `template<typename _CharT, bool _Intl> virtual string_type std::moneypunct<_CharT, _Intl>::do_negative_sign () const`
`[inline], [protected], [virtual]`

Return negative sign string.

This function returns a `string_type` to use as a sign for negative amounts. This function is a hook for derived classes to change the value returned.

See also

`negative_sign()` for details.

Returns

string_type representing a negative sign.

Definition at line 1321 of file `locale_facets_nonio.h`.

5.879.4.9 `template<typename _CharT, bool _Intl> virtual pattern std::moneypunct<_CharT, _Intl>::do_pos_format () const`
`[inline], [protected], [virtual]`

Return pattern for money values.

This function returns a pattern describing the formatting of a positive valued money amount. This function is a hook for derived classes to change the value returned.

See also

`pos_format()` for details.

Returns

Pattern for money values.

Definition at line 1349 of file `locale_facets_nonio.h`.

```
5.879.4.10 template<typename _CharT, bool _Intl> virtual string_type std::moneypunct< _CharT, _Intl >::do_positive_sign (
    ) const    [inline], [protected], [virtual]
```

Return positive sign string.

This function returns a `string_type` to use as a sign for positive amounts. This function is a hook for derived classes to change the value returned.

See also

`positive_sign()` for details.

Returns

string_type representing a positive sign.

Definition at line 1308 of file `locale_facets_nonio.h`.

```
5.879.4.11 template<typename _CharT, bool _Intl> virtual char_type std::moneypunct< _CharT, _Intl >::do_thousands_sep (
    ) const    [inline], [protected], [virtual]
```

Return thousands separator character.

Returns a `char_type` to use as a thousands separator. This function is a hook for derived classes to change the value returned.

Returns

char_type representing a thousands separator.

Definition at line 1269 of file `locale_facets_nonio.h`.

```
5.879.4.12 template<typename _CharT, bool _Intl> int std::moneypunct< _CharT, _Intl >::frac_digits ( ) const    [inline]
```

Return number of digits in fraction.

This function returns the exact number of digits that make up the fractional part of a money amount. It does so by returning `returning moneypunct<char_type>::do_frac_digits()`.

The fractional part of a money amount is optional. But if it is present, there must be `frac_digits()` digits.

Returns

Number of digits in amount fraction.

Definition at line 1199 of file `locale_facets_nonio.h`.

5.879.4.13 `template<typename _CharT, bool _Intl> string std::moneypunct<_CharT, _Intl>::grouping () const`
`[inline]`

Return grouping specification.

This function returns a string representing groupings for the integer part of an amount. Groupings indicate where thousands separators should be inserted.

Each char in the return string is interpreted as an integer rather than a character. These numbers represent the number of digits in a group. The first char in the string represents the number of digits in the least significant group. If a char is negative, it indicates an unlimited number of digits for the group. If more chars from the string are required to group a number, the last char is used repeatedly.

For example, if the `grouping()` returns `\003\002` and is applied to the number 123456789, this corresponds to 12,34,56,789. Note that if the string was 32, this would put more than 50 digits into the least significant group if the character set is ASCII.

The string is returned by calling `moneypunct<char_type>::do_grouping()`.

Returns

string representing grouping specification.

Definition at line 1136 of file `locale_facets_nonio.h`.

5.879.4.14 `template<typename _CharT, bool _Intl> pattern std::moneypunct<_CharT, _Intl>::neg_format () const`
`[inline]`

Return pattern for money values.

This function returns a pattern describing the formatting of a positive or negative valued money amount. It does so by returning `moneypunct<char_type>::do_pos_format()` or `moneypunct<char_type>::do_neg_format()`.

The pattern has 4 fields describing the ordering of symbol, sign, value, and none or space. There must be one of each in the pattern. The none and space enums may not appear in the first field and space may not appear in the final field.

The parts of a money string must appear in the order indicated by the fields of the pattern. The symbol field indicates that the value of `curr_symbol()` may be present. The sign field indicates that the value of `positive_sign()` or `negative_sign()` must be present. The value field indicates that the absolute value of the money amount is present. none indicates 0 or more whitespace characters, except at the end, where it permits no whitespace. space indicates that 1 or more whitespace characters must be present.

For example, for the US locale and `pos_format()` pattern {symbol,sign,value,none}, `curr_symbol() == '$'` `positive_sign() == '+'`, and value 10.01, and options set to force the symbol, the corresponding string is `$+10.01`.

Returns

Pattern for money values.

Definition at line 1239 of file `locale_facets_nonio.h`.

5.879.4.15 `template<typename _CharT, bool _Intl> string_type std::moneypunct< _CharT, _Intl >::negative_sign () const`
`[inline]`

Return negative sign string.

This function returns a `string_type` to use as a sign for negative amounts. It does so by returning `moneypunct<char_type>::do_negative_sign()`.

If the return value contains more than one character, the first character appears in the position indicated by `neg_format()` and the remainder appear at the end of the formatted string.

Returns

string_type representing a negative sign.

Definition at line 1183 of file `locale_facets_nonio.h`.

5.879.4.16 `template<typename _CharT, bool _Intl> pattern std::moneypunct< _CharT, _Intl >::pos_format () const`
`[inline]`

Return pattern for money values.

This function returns a pattern describing the formatting of a positive or negative valued money amount. It does so by returning `moneypunct<char_type>::do_pos_format()` or `moneypunct<char_type>::do_neg_format()`.

The pattern has 4 fields describing the ordering of symbol, sign, value, and none or space. There must be one of each in the pattern. The none and space enums may not appear in the first field and space may not appear in the final field.

The parts of a money string must appear in the order indicated by the fields of the pattern. The symbol field indicates that the value of `curr_symbol()` may be present. The sign field indicates that the value of `positive_sign()` or `negative_sign()` must be present. The value field indicates that the absolute value of the money amount is present. none indicates 0 or more whitespace characters, except at the end, where it permits no whitespace. space indicates that 1 or more whitespace characters must be present.

For example, for the US locale and `pos_format()` pattern {symbol,sign,value,none}, `curr_symbol() == '$'` `positive_sign() == '+'`, and value 10.01, and options set to force the symbol, the corresponding string is \$+10.01.

Returns

Pattern for money values.

Definition at line 1235 of file `locale_facets_nonio.h`.

5.879.4.17 `template<typename _CharT, bool _Intl> string_type std::moneypunct< _CharT, _Intl >::positive_sign () const`
`[inline]`

Return positive sign string.

This function returns a `string_type` to use as a sign for positive amounts. It does so by returning `moneypunct<char_type>::do_positive_sign()`.

If the return value contains more than one character, the first character appears in the position indicated by `pos_format()` and the remainder appear at the end of the formatted string.

Returns

string_type representing a positive sign.

Definition at line 1166 of file `locale_facets_nonio.h`.

5.879.4.18 `template<typename _CharT, bool _Intl> char_type std::moneypunct<_CharT, _Intl>::thousands_sep () const`
`[inline]`

Return thousands separator character.

This function returns a `char_type` to use as a thousands separator. It does so by returning `moneypunct<char_type>::do_thousands_sep()`.

Returns

`char_type` representing a thousands separator.

Definition at line 1106 of file `locale_facets_nonio.h`.

5.879.5 Member Data Documentation

5.879.5.1 `template<typename _CharT, bool _Intl> locale::id std::moneypunct<_CharT, _Intl>::id` `[static]`

Numpunct facet id.

Definition at line 1041 of file `locale_facets_nonio.h`.

5.879.5.2 `template<typename _CharT, bool _Intl> const bool std::moneypunct<_CharT, _Intl>::intl` `[static]`

This value is provided by the standard, but no reason for its existence.

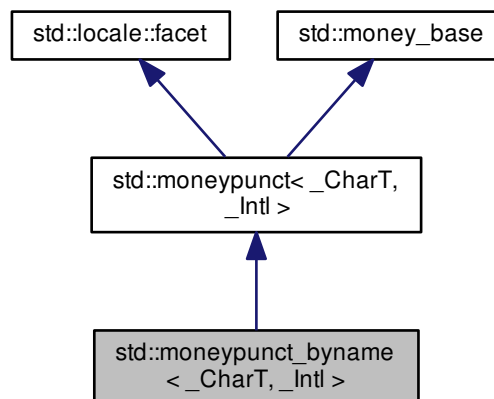
Definition at line 1039 of file `locale_facets_nonio.h`.

The documentation for this class was generated from the following file:

- [locale_facets_nonio.h](#)

5.880 std::moneypunct_byname<_CharT, _Intl> Class Template Reference

Inheritance diagram for `std::moneypunct_byname<_CharT, _Intl>`:



Public Types

- enum { **_S_minus**, **_S_zero**, **_S_end** }
- typedef __moneypunct_cache< _CharT, _Intl > **__cache_type**
- typedef _CharT **char_type**
- enum **part** {
 none, **space**, **symbol**, **sign**,
 value }
- typedef [basic_string](#)< _CharT > **string_type**

Public Member Functions

- **moneypunct_byname** (const char *__s, size_t __refs=0)
- **moneypunct_byname** (const [string](#) &__s, size_t __refs=0)
- [string_type](#) **curr_symbol** () const
- [char_type](#) **decimal_point** () const
- int **frac_digits** () const
- [string](#) **grouping** () const
- [string_type](#) **negative_sign** () const
- [string_type](#) **positive_sign** () const
- [char_type](#) **thousands_sep** () const

- pattern [pos_format](#) () const
- pattern [neg_format](#) () const

Static Public Member Functions

- static pattern **_S_construct_pattern** (char __precedes, char __space, char __posn) throw ()

Static Public Attributes

- static const char * **_S_atoms**
- static const pattern **_S_default_pattern**
- static [locale::id](#) **id**
- static const bool **intl**

Protected Member Functions

- void **_M_initialize_moneypunct** (__c_locale __cloc=0, const char *__name=0)
- template<>
void **_M_initialize_moneypunct** (__c_locale, const char *)
- template<>
void **_M_initialize_moneypunct** (__c_locale, const char *)
- template<>
void **_M_initialize_moneypunct** (__c_locale, const char *)
- template<>
void **_M_initialize_moneypunct** (__c_locale, const char *)
- virtual [string_type](#) **do_curr_symbol** () const
- virtual [char_type](#) **do_decimal_point** () const
- virtual int **do_frac_digits** () const
- virtual [string](#) **do_grouping** () const
- virtual pattern **do_neg_format** () const
- virtual [string_type](#) **do_negative_sign** () const
- virtual pattern **do_pos_format** () const
- virtual [string_type](#) **do_positive_sign** () const
- virtual [char_type](#) **do_thousands_sep** () const

Static Protected Member Functions

- static __c_locale **_S_clone_c_locale** (__c_locale &__cloc) throw ()
- static void **_S_create_c_locale** (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale &__cloc)
- static __c_locale **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static __c_locale **_S_lc_ctype_c_locale** (__c_locale __cloc, const char *__s)

5.880.1 Detailed Description

```
template<typename _CharT, bool _Intl>
class std::moneypunct_byname<_CharT, _Intl>
```

class moneypunct_byname [22.2.6.4].

Definition at line 1412 of file locale_facets_nonio.h.

5.880.2 Member Function Documentation

5.880.2.1 template<typename _CharT, bool _Intl> **string_type** std::moneypunct<_CharT, _Intl>::curr_symbol () const
[inline], [inherited]

Return currency symbol string.

This function returns a [string_type](#) to use as a currency symbol. It does so by returning returning moneypunct<char↔_type>::do_curr_symbol().

Returns

string_type representing a currency symbol.

Definition at line 1149 of file locale_facets_nonio.h.

5.880.2.2 `template<typename _CharT, bool _Intl> char_type std::moneypunct<_CharT, _Intl >::decimal_point () const`
`[inline], [inherited]`

Return decimal point character.

This function returns a `char_type` to use as a decimal point. It does so by returning `moneypunct<char_type>::do_decimal_point()`.

Returns

char_type representing a decimal point.

Definition at line 1093 of file `locale_facets_nonio.h`.

5.880.2.3 `template<typename _CharT, bool _Intl> virtual string_type std::moneypunct<_CharT, _Intl >::do_curr_symbol () const`
`[inline], [protected], [virtual], [inherited]`

Return currency symbol string.

This function returns a `string_type` to use as a currency symbol. This function is a hook for derived classes to change the value returned.

See also

`curr_symbol()` for details.

Returns

string_type representing a currency symbol.

Definition at line 1295 of file `locale_facets_nonio.h`.

5.880.2.4 `template<typename _CharT, bool _Intl> virtual char_type std::moneypunct<_CharT, _Intl >::do_decimal_point () const`
`[inline], [protected], [virtual], [inherited]`

Return decimal point character.

Returns a `char_type` to use as a decimal point. This function is a hook for derived classes to change the value returned.

Returns

char_type representing a decimal point.

Definition at line 1257 of file `locale_facets_nonio.h`.

5.880.2.5 `template<typename _CharT, bool _Intl> virtual int std::moneypunct<_CharT, _Intl>::do_frac_digits () const`
`[inline], [protected], [virtual], [inherited]`

Return number of digits in fraction.

This function returns the exact number of digits that make up the fractional part of a money amount. This function is a hook for derived classes to change the value returned.

See also

`frac_digits()` for details.

Returns

Number of digits in amount fraction.

Definition at line 1335 of file `locale_facets_nonio.h`.

5.880.2.6 `template<typename _CharT, bool _Intl> virtual string std::moneypunct<_CharT, _Intl>::do_grouping () const`
`[inline], [protected], [virtual], [inherited]`

Return grouping specification.

Returns a string representing groupings for the integer part of a number. This function is a hook for derived classes to change the value returned.

See also

`grouping()` for details.

Returns

String representing grouping specification.

Definition at line 1282 of file `locale_facets_nonio.h`.

5.880.2.7 `template<typename _CharT, bool _Intl> virtual pattern std::moneypunct<_CharT, _Intl>::do_neg_format () const`
`[inline], [protected], [virtual], [inherited]`

Return pattern for money values.

This function returns a pattern describing the formatting of a negative valued money amount. This function is a hook for derived classes to change the value returned.

See also

`neg_format()` for details.

Returns

Pattern for money values.

Definition at line 1363 of file `locale_facets_nonio.h`.

References `std::moneypunct<_CharT, _Intl>::~~moneypunct()`.

5.880.2.8 `template<typename _CharT, bool _Intl> virtual string_type std::moneypunct<_CharT, _Intl>::do_negative_sign () const` `[inline], [protected], [virtual], [inherited]`

Return negative sign string.

This function returns a `string_type` to use as a sign for negative amounts. This function is a hook for derived classes to change the value returned.

See also

`negative_sign()` for details.

Returns

string_type representing a negative sign.

Definition at line 1321 of file `locale_facets_nonio.h`.

5.880.2.9 `template<typename _CharT, bool _Intl> virtual pattern std::moneypunct<_CharT, _Intl>::do_pos_format () const` `[inline], [protected], [virtual], [inherited]`

Return pattern for money values.

This function returns a pattern describing the formatting of a positive valued money amount. This function is a hook for derived classes to change the value returned.

See also

`pos_format()` for details.

Returns

Pattern for money values.

Definition at line 1349 of file `locale_facets_nonio.h`.

5.880.2.10 `template<typename _CharT, bool _Intl> virtual string_type std::moneypunct<_CharT, _Intl>::do_positive_sign () const` `[inline], [protected], [virtual], [inherited]`

Return positive sign string.

This function returns a `string_type` to use as a sign for positive amounts. This function is a hook for derived classes to change the value returned.

See also

`positive_sign()` for details.

Returns

string_type representing a positive sign.

Definition at line 1308 of file `locale_facets_nonio.h`.

```
5.880.2.11 template<typename _CharT, bool _Intl> virtual char_type std::moneypunct<_CharT, _Intl >::do_thousands_sep (
    ) const    [inline], [protected], [virtual], [inherited]
```

Return thousands separator character.

Returns a char_type to use as a thousands separator. This function is a hook for derived classes to change the value returned.

Returns

char_type representing a thousands separator.

Definition at line 1269 of file locale_facets_nonio.h.

```
5.880.2.12 template<typename _CharT, bool _Intl> int std::moneypunct<_CharT, _Intl >::frac_digits (    ) const
    [inline], [inherited]
```

Return number of digits in fraction.

This function returns the exact number of digits that make up the fractional part of a money amount. It does so by returning returning moneypunct<char_type>::do_frac_digits().

The fractional part of a money amount is optional. But if it is present, there must be frac_digits() digits.

Returns

Number of digits in amount fraction.

Definition at line 1199 of file locale_facets_nonio.h.

```
5.880.2.13 template<typename _CharT, bool _Intl> string std::moneypunct<_CharT, _Intl >::grouping (    ) const
    [inline], [inherited]
```

Return grouping specification.

This function returns a string representing groupings for the integer part of an amount. Groupings indicate where thousands separators should be inserted.

Each char in the return string is interpret as an integer rather than a character. These numbers represent the number of digits in a group. The first char in the string represents the number of digits in the least significant group. If a char is negative, it indicates an unlimited number of digits for the group. If more chars from the string are required to group a number, the last char is used repeatedly.

For example, if the grouping() returns \003\002 and is applied to the number 123456789, this corresponds to 12,34,56,789. Note that if the string was 32, this would put more than 50 digits into the least significant group if the character set is ASCII.

The string is returned by calling moneypunct<char_type>::do_grouping().

Returns

string representing grouping specification.

Definition at line 1136 of file locale_facets_nonio.h.

5.880.2.14 `template<typename _CharT, bool _Intl> pattern std::moneypunct<_CharT, _Intl>::neg_format () const`
`[inline], [inherited]`

Return pattern for money values.

This function returns a pattern describing the formatting of a positive or negative valued money amount. It does so by returning `moneypunct<char_type>::do_pos_format()` or `moneypunct<char_type>::do_neg_format()`.

The pattern has 4 fields describing the ordering of symbol, sign, value, and none or space. There must be one of each in the pattern. The none and space enums may not appear in the first field and space may not appear in the final field.

The parts of a money string must appear in the order indicated by the fields of the pattern. The symbol field indicates that the value of `curr_symbol()` may be present. The sign field indicates that the value of `positive_sign()` or `negative_sign()` must be present. The value field indicates that the absolute value of the money amount is present. none indicates 0 or more whitespace characters, except at the end, where it permits no whitespace. space indicates that 1 or more whitespace characters must be present.

For example, for the US locale and `pos_format()` pattern {symbol,sign,value,none}, `curr_symbol() == '$'` `positive_sign() == '+'`, and value 10.01, and options set to force the symbol, the corresponding string is `$+10.01`.

Returns

Pattern for money values.

Definition at line 1239 of file `locale_facets_nonio.h`.

5.880.2.15 `template<typename _CharT, bool _Intl> string_type std::moneypunct<_CharT, _Intl>::negative_sign () const`
`[inline], [inherited]`

Return negative sign string.

This function returns a `string_type` to use as a sign for negative amounts. It does so by returning `moneypunct<char_type>::do_negative_sign()`.

If the return value contains more than one character, the first character appears in the position indicated by `neg_format()` and the remainder appear at the end of the formatted string.

Returns

string_type representing a negative sign.

Definition at line 1183 of file `locale_facets_nonio.h`.

5.880.2.16 `template<typename _CharT, bool _Intl> pattern std::moneypunct<_CharT, _Intl>::pos_format () const`
`[inline], [inherited]`

Return pattern for money values.

This function returns a pattern describing the formatting of a positive or negative valued money amount. It does so by returning `moneypunct<char_type>::do_pos_format()` or `moneypunct<char_type>::do_neg_format()`.

The pattern has 4 fields describing the ordering of symbol, sign, value, and none or space. There must be one of each in the pattern. The none and space enums may not appear in the first field and space may not appear in the final field.

The parts of a money string must appear in the order indicated by the fields of the pattern. The symbol field indicates that the value of `curr_symbol()` may be present. The sign field indicates that the value of `positive_sign()` or `negative_sign()` must be present. The value field indicates that the absolute value of the money amount is present. none indicates 0 or more whitespace characters, except at the end, where it permits no whitespace. space indicates that 1 or more whitespace characters must be present.

For example, for the US locale and `pos_format()` pattern {symbol,sign,value,none}, `curr_symbol() == '$'` `positive_sign() == '+'`, and value 10.01, and options set to force the symbol, the corresponding string is \$+10.01.

Returns

Pattern for money values.

Definition at line 1235 of file `locale_facets_nonio.h`.

5.880.2.17 `template<typename _CharT, bool _Intl> string_type std::moneypunct<_CharT, _Intl>::positive_sign () const`
`[inline], [inherited]`

Return positive sign string.

This function returns a `string_type` to use as a sign for positive amounts. It does so by returning `moneypunct<char_type>::do_positive_sign()`.

If the return value contains more than one character, the first character appears in the position indicated by `pos_format()` and the remainder appear at the end of the formatted string.

Returns

string_type representing a positive sign.

Definition at line 1166 of file `locale_facets_nonio.h`.

5.880.2.18 `template<typename _CharT, bool _Intl> char_type std::moneypunct<_CharT, _Intl>::thousands_sep () const`
`[inline], [inherited]`

Return thousands separator character.

This function returns a `char_type` to use as a thousands separator. It does so by returning `moneypunct<char_type>::do_thousands_sep()`.

Returns

`char_type` representing a thousands separator.

Definition at line 1106 of file `locale_facets_nonio.h`.

5.880.3 Member Data Documentation

5.880.3.1 `template<typename _CharT, bool _Intl> locale::id std::moneypunct<_CharT, _Intl>::id [static],
[inherited]`

Numpunct facet id.

Definition at line 1041 of file locale_facets_nonio.h.

The documentation for this class was generated from the following file:

- [locale_facets_nonio.h](#)

5.881 `std::move_iterator<_Iterator>` Class Template Reference

Public Types

- typedef `__traits_type::difference_type` **difference_type**
- typedef `__traits_type::iterator_category` **iterator_category**
- typedef `_Iterator` **iterator_type**
- typedef `_Iterator` **pointer**
- typedef conditional< [is_reference](#)< `__base_ref`>::value, typename remove_reference< `__base_ref`>::type &&, `__base_ref`>::type **reference**
- typedef `__traits_type::value_type` **value_type**

Public Member Functions

- **move_iterator** (`iterator_type __i`)
- template<typename `_Iter`>
 move_iterator (const [move_iterator](#)< `_Iter`> &`__i`)
- `iterator_type` **base** () const
- reference **operator*** () const
- [move_iterator](#) **operator+** (`difference_type __n`) const
- [move_iterator](#) & **operator++** ()
- [move_iterator](#) **operator++** (`int`)
- [move_iterator](#) & **operator+=** (`difference_type __n`)
- [move_iterator](#) **operator-** (`difference_type __n`) const
- [move_iterator](#) & **operator--** ()
- [move_iterator](#) **operator--** (`int`)
- [move_iterator](#) & **operator-=** (`difference_type __n`)
- pointer **operator->** () const
- reference **operator[]** (`difference_type __n`) const

Protected Types

- typedef `__traits_type::reference` **base_ref**
- typedef `iterator_traits<_Iterator>` **__traits_type**

Protected Attributes

- `_Iterator _M_current`

5.881.1 Detailed Description

```
template<typename _Iterator>
class std::move_iterator< _Iterator >
```

Class template `move_iterator` is an iterator adapter with the same behavior as the underlying iterator except that its dereference operator implicitly converts the value returned by the underlying iterator's dereference operator to an rvalue reference. Some generic algorithms can be called with move iterators to replace copying with moving.

Definition at line 1007 of file `bits/stl_iterator.h`.

The documentation for this class was generated from the following file:

- [bits/stl_iterator.h](#)

5.882 std::multimap< _Key, _Tp, _Compare, _Alloc > Class Template Reference

Public Types

- typedef `_Alloc` **allocator_type**
- typedef `_Rep_type::const_iterator` **const_iterator**
- typedef `_Alloc_traits::const_pointer` **const_pointer**
- typedef `_Alloc_traits::const_reference` **const_reference**
- typedef `_Rep_type::const_reverse_iterator` **const_reverse_iterator**
- typedef `_Rep_type::difference_type` **difference_type**
- typedef `_Rep_type::iterator` **iterator**
- typedef `_Compare` **key_compare**
- typedef `_Key` **key_type**
- typedef `_Tp` **mapped_type**
- typedef `_Alloc_traits::pointer` **pointer**
- typedef `_Alloc_traits::reference` **reference**
- typedef `_Rep_type::reverse_iterator` **reverse_iterator**
- typedef `_Rep_type::size_type` **size_type**
- typedef `std::pair< const _Key, _Tp >` **value_type**

Public Member Functions

- [multimap](#) () noexcept(*/*conditional */*)
- [multimap](#) (const [_Compare](#) &__comp, const [allocator_type](#) &__a=allocator_type())
- [multimap](#) (const [multimap](#) &__x)
- [multimap](#) ([multimap](#) &&__x) noexcept(is_nothrow_copy_constructible< [_Compare](#) >::value)
- [multimap](#) ([initializer_list](#)< [value_type](#) > __l, const [_Compare](#) &__comp=[_Compare](#)(), const [allocator_type](#) &__a=allocator_type())
- [multimap](#) (const [allocator_type](#) &__a)
- [multimap](#) (const [multimap](#) &__m, const [allocator_type](#) &__a)
- [multimap](#) ([multimap](#) &&__m, const [allocator_type](#) &__a) noexcept(is_nothrow_copy_constructible< [_Compare](#) >::value && [Alloc_traits](#)::S_always_equal())
- [multimap](#) ([initializer_list](#)< [value_type](#) > __l, const [allocator_type](#) &__a)
- template<typename [_InputIterator](#) >
[multimap](#) ([_InputIterator](#) __first, [_InputIterator](#) __last, const [allocator_type](#) &__a)
- template<typename [_InputIterator](#) >
[multimap](#) ([_InputIterator](#) __first, [_InputIterator](#) __last)
- template<typename [_InputIterator](#) >
[multimap](#) ([_InputIterator](#) __first, [_InputIterator](#) __last, const [_Compare](#) &__comp, const [allocator_type](#) &__a=allocator_type())
- iterator [begin](#) () noexcept
- const_iterator [begin](#) () const noexcept
- const_iterator [cbegin](#) () const noexcept
- const_iterator [cend](#) () const noexcept
- void [clear](#) () noexcept
- const_reverse_iterator [crbegin](#) () const noexcept
- const_reverse_iterator [crend](#) () const noexcept
- template<typename... [_Args](#)>
iterator [emplace](#) ([_Args](#) &&...__args)
- template<typename... [_Args](#)>
iterator [emplace_hint](#) (const_iterator __pos, [_Args](#) &&...__args)
- bool [empty](#) () const noexcept
- iterator [end](#) () noexcept
- const_iterator [end](#) () const noexcept
- iterator [erase](#) (const_iterator __position)
- [_GLIBCXX_ABI_TAG_CXX11](#) iterator **erase** (iterator __position)
- size_type [erase](#) (const key_type &__x)
- iterator [erase](#) (const_iterator __first, const_iterator __last)
- [allocator_type](#) [get_allocator](#) () const noexcept
- iterator [insert](#) (const [value_type](#) &__x)
- template<typename [_Pair](#) , typename = typename std::enable_if<std::is_constructible<[value_type](#), [_Pair](#)&&>::value>::type>
iterator **insert** ([_Pair](#) &&__x)
- iterator [insert](#) (const_iterator __position, const [value_type](#) &__x)
- template<typename [_Pair](#) , typename = typename std::enable_if<std::is_constructible<[value_type](#), [_Pair](#)&&>::value>::type>
iterator **insert** (const_iterator __position, [_Pair](#) &&__x)
- template<typename [_InputIterator](#) >
void [insert](#) ([_InputIterator](#) __first, [_InputIterator](#) __last)
- void [insert](#) ([initializer_list](#)< [value_type](#) > __l)
- key_compare [key_comp](#) () const
- size_type [max_size](#) () const noexcept
- [multimap](#) & [operator=](#) (const [multimap](#) &__x)

- `multimap` & `operator=` (`multimap` &&)=default
 - `multimap` & `operator=` (`initializer_list`< `value_type` > __l)
 - `reverse_iterator` `rbegin` () noexcept
 - `const_reverse_iterator` `rbegin` () const noexcept
 - `reverse_iterator` `rend` () noexcept
 - `const_reverse_iterator` `rend` () const noexcept
 - `size_type` `size` () const noexcept
 - void `swap` (`multimap` &__x) noexcept(*/*conditional */*)
 - `value_compare` `value_comp` () const
-
- iterator `find` (const `key_type` &__x)
 - template<typename `_Kt` >
auto `find` (const `_Kt` &__x) -> decltype(`_M.t._M_find_tr`(__x))
-
- const_iterator `find` (const `key_type` &__x) const
 - template<typename `_Kt` >
auto `find` (const `_Kt` &__x) const -> decltype(`_M.t._M_find_tr`(__x))
-
- `size_type` `count` (const `key_type` &__x) const
 - template<typename `_Kt` >
auto `count` (const `_Kt` &__x) const -> decltype(`_M.t._M_count_tr`(__x))
-
- iterator `lower_bound` (const `key_type` &__x)
 - template<typename `_Kt` >
auto `lower_bound` (const `_Kt` &__x) -> decltype(`_M.t._M_lower_bound_tr`(__x))
-
- const_iterator `lower_bound` (const `key_type` &__x) const
 - template<typename `_Kt` >
auto `lower_bound` (const `_Kt` &__x) const -> decltype(`_M.t._M_lower_bound_tr`(__x))
-
- iterator `upper_bound` (const `key_type` &__x)
 - template<typename `_Kt` >
auto `upper_bound` (const `_Kt` &__x) -> decltype(`_M.t._M_upper_bound_tr`(__x))
-
- const_iterator `upper_bound` (const `key_type` &__x) const
 - template<typename `_Kt` >
auto `upper_bound` (const `_Kt` &__x) const -> decltype(`_M.t._M_upper_bound_tr`(__x))
-
- `std::pair`< iterator, iterator > `equal_range` (const `key_type` &__x)
 - template<typename `_Kt` >
auto `equal_range` (const `_Kt` &__x) -> decltype(`_M.t._M_equal_range_tr`(__x))
-
- `std::pair`< const_iterator, const_iterator > `equal_range` (const `key_type` &__x) const
 - template<typename `_Kt` >
auto `equal_range` (const `_Kt` &__x) const -> decltype(`_M.t._M_equal_range_tr`(__x))

Friends

- `template<typename _K1, typename _T1, typename _C1, typename _A1 >`
`bool operator< (const multimap< _K1, _T1, _C1, _A1 > &, const multimap< _K1, _T1, _C1, _A1 > &)`
- `template<typename _K1, typename _T1, typename _C1, typename _A1 >`
`bool operator== (const multimap< _K1, _T1, _C1, _A1 > &, const multimap< _K1, _T1, _C1, _A1 > &)`

5.882.1 Detailed Description

```
template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<std::pair<const
_Key, _Tp>>>
class std::multimap< _Key, _Tp, _Compare, _Alloc >
```

A standard container made up of (key,value) pairs, which can be retrieved based on a key, in logarithmic time.

Template Parameters

<code>_Key</code>	Type of key objects.
<code>_Tp</code>	Type of mapped objects.
<code>_Compare</code>	Comparison function object type, defaults to <code>less<_Key></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<pair<const _Key, _Tp>></code> .

Meets the requirements of a [container](#), a [reversible container](#), and an [associative container](#) (using equivalent keys). For a `multimap<Key, T>` the `key_type` is `Key`, the `mapped_type` is `T`, and the `value_type` is `std::pair<const Key, T>`.

Multimaps support bidirectional iterators.

The private tree data is declared exactly the same way for `map` and `multimap`; the distinction is made entirely in how the tree functions are called (`*_unique` versus `*_equal`, same as the standard).

Definition at line 95 of file `stl_multimap.h`.

5.882.2 Constructor & Destructor Documentation

5.882.2.1 `template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =`
`std::allocator<std::pair<const _Key, _Tp>>> std::multimap< _Key, _Tp, _Compare, _Alloc >::multimap ()`
`[inline], [noexcept]`

Default constructor creates no elements.

Definition at line 160 of file `stl_multimap.h`.

5.882.2.2 `template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =`
`std::allocator<std::pair<const _Key, _Tp>>> std::multimap< _Key, _Tp, _Compare, _Alloc >::multimap (const`
`_Compare & __comp, const allocator_type & __a = allocator_type()) [inline], [explicit]`

Creates a `multimap` with no elements.

Parameters

<code>__comp</code>	A comparison object.
<code>__a</code>	An allocator object.

Definition at line 172 of file `stl_multimap.h`.

5.882.2.3 `template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<std::pair<const _Key, _Tp>>> std::multimap<_Key, _Tp, _Compare, _Alloc>::multimap (const multimap<_Key, _Tp, _Compare, _Alloc> & __x) [inline]`

Multimap copy constructor.

Parameters

<code>__x</code>	A multimap of identical element and allocator types.
------------------	--

The newly-created multimap uses a copy of the allocation object used by `__x`.

Definition at line 183 of file `stl_multimap.h`.

5.882.2.4 `template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<std::pair<const _Key, _Tp>>> std::multimap<_Key, _Tp, _Compare, _Alloc>::multimap (multimap<_Key, _Tp, _Compare, _Alloc> && __x) [inline], [noexcept]`

Multimap move constructor.

Parameters

<code>__x</code>	A multimap of identical element and allocator types.
------------------	--

The newly-created multimap contains the exact contents of `__x`. The contents of `__x` are a valid, but unspecified multimap.

Definition at line 194 of file `stl_multimap.h`.

5.882.2.5 `template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<std::pair<const _Key, _Tp>>> std::multimap<_Key, _Tp, _Compare, _Alloc>::multimap (initializer_list<value_type> & __l, const _Compare & __comp = _Compare(), const allocator_type & __a = allocator_type()) [inline]`

Builds a multimap from an `initializer_list`.

Parameters

<code>__l</code>	An <code>initializer_list</code> .
<code>__comp</code>	A comparison functor.
<code>__a</code>	An allocator object.

Create a multimap consisting of copies of the elements from the `initializer_list`. This is linear in N if the list is already sorted, and $N \log N$ otherwise (where N is `__l.size()`).

Definition at line 208 of file `stl_multimap.h`.

```
5.882.2.6  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp>>> std::multimap<_Key, _Tp, _Compare, _Alloc>::multimap ( const
            allocator_type & __a ) [inline], [explicit]
```

Allocator-extended default constructor.

Definition at line 216 of file `stl_multimap.h`.

```
5.882.2.7  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp>>> std::multimap<_Key, _Tp, _Compare, _Alloc>::multimap ( const
            multimap<_Key, _Tp, _Compare, _Alloc> & __m, const allocator_type & __a ) [inline]
```

Allocator-extended copy constructor.

Definition at line 220 of file `stl_multimap.h`.

```
5.882.2.8  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp>>> std::multimap<_Key, _Tp, _Compare, _Alloc>::multimap (
            multimap<_Key, _Tp, _Compare, _Alloc> && __m, const allocator_type & __a ) [inline], [noexcept]
```

Allocator-extended move constructor.

Definition at line 224 of file `stl_multimap.h`.

```
5.882.2.9  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp>>> std::multimap<_Key, _Tp, _Compare, _Alloc>::multimap (
            initializer_list< value_type > __l, const allocator_type & __a ) [inline]
```

Allocator-extended initializer-list constructor.

Definition at line 230 of file `stl_multimap.h`.

```
5.882.2.10 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp>>> template<typename _InputIterator> std::multimap<_Key,
            _Tp, _Compare, _Alloc>::multimap ( _InputIterator __first, _InputIterator __last, const allocator_type & __a )
            [inline]
```

Allocator-extended range constructor.

Definition at line 236 of file `stl_multimap.h`.

```
5.882.2.11 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp>>> template<typename _InputIterator> std::multimap<_Key, _Tp,
            _Compare, _Alloc>::multimap ( _InputIterator __first, _InputIterator __last ) [inline]
```

Builds a multimap from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Create a multimap consisting of copies of the elements from [`__first`,`__last`). This is linear in N if the range is already sorted, and NlogN otherwise (where N is distance(`__first`,`__last`)).

Definition at line 252 of file `stl_multimap.h`.

```
5.882.2.12 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> template<typename _InputIterator > std::multimap< _Key, _Tp,
_Compare, _Alloc >::multimap ( _InputIterator __first, _InputIterator __last, const _Compare & __comp, const
allocator_type & __a = allocator_type() ) [inline]
```

Builds a multimap from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__comp</code>	A comparison functor.
<code>__a</code>	An allocator object.

Create a multimap consisting of copies of the elements from [`__first`,`__last`). This is linear in N if the range is already sorted, and NlogN otherwise (where N is distance(`__first`,`__last`)).

Definition at line 268 of file `stl_multimap.h`.

5.882.3 Member Function Documentation

```
5.882.3.1 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::begin ( )
[inline], [noexcept]
```

Returns a read/write iterator that points to the first pair in the multimap. Iteration is done in ascending order according to the keys.

Definition at line 333 of file `stl_multimap.h`.

```
5.882.3.2 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> const_iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::begin
( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first pair in the multimap. Iteration is done in ascending order according to the keys.

Definition at line 342 of file `stl_multimap.h`.

```
5.882.3.3 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> const_iterator std::multimap<_Key, _Tp, _Compare, _Alloc
>::cbegin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first pair in the multimap. Iteration is done in ascending order according to the keys.

Definition at line 406 of file stl_multimap.h.

```
5.882.3.4 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> const_iterator std::multimap<_Key, _Tp, _Compare, _Alloc >::cend (
) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last pair in the multimap. Iteration is done in ascending order according to the keys.

Definition at line 415 of file stl_multimap.h.

```
5.882.3.5 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> void std::multimap<_Key, _Tp, _Compare, _Alloc >::clear ( )
[inline], [noexcept]
```

Erases all elements in a multimap. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 717 of file stl_multimap.h.

```
5.882.3.6 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> size_type std::multimap<_Key, _Tp, _Compare, _Alloc >::count (
const key_type & __x ) const [inline]
```

Finds the number of elements with given key.

Parameters

<code>__x</code>	Key of (key, value) pairs to be located.
------------------	--

Returns

Number of elements with specified key.

Definition at line 794 of file stl_multimap.h.

```
5.882.3.7 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> template<typename _Kt > auto std::multimap<_Key, _Tp,
_Compare, _Alloc >::count ( const _Kt & __x ) const -> decltype(_M_t._M_count_tr(__x)) [inline]
```

Finds the number of elements with given key.

Parameters

<code>_↔</code>	Key of (key, value) pairs to be located.
<code>_X</code>	

Returns

Number of elements with specified key.

Definition at line 800 of file stl_multimap.h.

```
5.882.3.8 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> const_reverse_iterator std::multimap<_Key, _Tp, _Compare,
_Alloc>::crbegin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last pair in the multimap. Iteration is done in descending order according to the keys.

Definition at line 424 of file stl_multimap.h.

```
5.882.3.9 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> const_reverse_iterator std::multimap<_Key, _Tp, _Compare,
_Alloc>::crend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to one before the first pair in the multimap. Iteration is done in descending order according to the keys.

Definition at line 433 of file stl_multimap.h.

```
5.882.3.10 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> template<typename... _Args> iterator std::multimap<_Key, _Tp,
_Compare, _Alloc>::emplace ( _Args &&... _args ) [inline]
```

Build and insert a std::pair into the multimap.

Parameters

<code>__args</code>	Arguments used to generate a new pair instance (see std::piecewise_construct for passing arguments to each part of the pair constructor).
---------------------	---

Returns

An iterator that points to the inserted (key,value) pair.

This function builds and inserts a (key, value) pair into the multimap. Contrary to a std::map the multimap does not rely on unique keys and thus multiple pairs with the same key can be inserted.

Insertion requires logarithmic time.

Definition at line 473 of file stl_multimap.h.


```
5.882.3.11 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> template<typename... _Args> iterator std::multimap<_Key, _Tp,
_Compare, _Alloc >::emplace_hint ( const_iterator __pos, _Args &&... __args ) [inline]
```

Builds and inserts a `std::pair` into the multimap.

Parameters

<code>__pos</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__args</code>	Arguments used to generate a new pair instance (see <code>std::piecewise_construct</code> for passing arguments to each part of the pair constructor).

Returns

An iterator that points to the inserted (key,value) pair.

This function inserts a (key, value) pair into the multimap. Contrary to a `std::map` the multimap does not rely on unique keys and thus multiple pairs with the same key can be inserted. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints

Insertion requires logarithmic time (if the hint is not taken).

Definition at line 500 of file `stl_multimap.h`.

```
5.882.3.12 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> bool std::multimap<_Key, _Tp, _Compare, _Alloc >::empty ( )
const [inline], [noexcept]
```

Returns true if the multimap is empty.

Definition at line 440 of file `stl_multimap.h`.

```
5.882.3.13 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> iterator std::multimap<_Key, _Tp, _Compare, _Alloc >::end ( )
[inline], [noexcept]
```

Returns a read/write iterator that points one past the last pair in the multimap. Iteration is done in ascending order according to the keys.

Definition at line 351 of file `stl_multimap.h`.

```
5.882.3.14 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> const_iterator std::multimap<_Key, _Tp, _Compare, _Alloc >::end (
) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last pair in the multimap. Iteration is done in ascending order according to the keys.

Definition at line 360 of file `stl_multimap.h`.

```
5.882.3.15 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> std::pair<iterator, iterator> std::multimap<_Key, _Tp, _Compare,
_Alloc >::equal_range ( const key_type & __x ) [inline]
```

Finds a subsequence matching given key.

Parameters

<code>_↔</code>	Key of (key, value) pairs to be located.
<code>_X</code>	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
              c.upper_bound(val))
```

(but is faster than making the calls separately).

Definition at line 910 of file stl_multimap.h.

```
5.882.3.16 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> template<typename _Kt > auto std::multimap< _Key, _Tp,
_Compare, _Alloc >::equal_range ( const _Kt & __x ) -> decltype(_M_t._M_equal_range_tr(__x)) [inline]
```

Finds a subsequence matching given key.

Parameters

<code>_↔</code>	Key of (key, value) pairs to be located.
<code>_X</code>	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
              c.upper_bound(val))
```

(but is faster than making the calls separately).

Definition at line 916 of file stl_multimap.h.

```
5.882.3.17 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> std::pair<const_iterator, const_iterator> std::multimap< _Key,
_Tp, _Compare, _Alloc >::equal_range ( const key_type & __x ) const [inline]
```

Finds a subsequence matching given key.

Parameters

<code>_↔</code>	Key of (key, value) pairs to be located.
<code>_X</code>	

Returns

Pair of read-only (constant) iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
               c.upper_bound(val))
```

(but is faster than making the calls separately).

Definition at line 937 of file stl_multimap.h.

```
5.882.3.18 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp> >> template<typename _Kt > auto std::multimap<_Key, _Tp,
            _Compare, _Alloc >::equal_range ( const _Kt & __x ) const-> decltype(_M_t.M_equal_range_tr(__x)) [inline]
```

Finds a subsequence matching given key.

Parameters

<code>_↔</code>	Key of (key, value) pairs to be located.
<code>_X</code>	

Returns

Pair of read-only (constant) iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
               c.upper_bound(val))
```

(but is faster than making the calls separately).

Definition at line 943 of file stl_multimap.h.

References `std::operator==()`.

```
5.882.3.19 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp> >> iterator std::multimap<_Key, _Tp, _Compare, _Alloc >::erase (
            const_iterator __position ) [inline]
```

Erases an element from a multimap.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following *position* prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from a multimap. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 614 of file `stl_multimap.h`.

Referenced by `std::multimap< _Key, _Tp, _Compare, _Alloc >::erase()`.

```
5.882.3.20 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >> size_type std::multimap< _Key, _Tp, _Compare, _Alloc >::erase (
const key_type & __x ) [inline]
```

Erases elements according to the provided key.

Parameters

<code>__key</code>	Key of element to be erased.
<code>__x</code>	

Returns

The number of elements erased.

This function erases all elements located by the given key from a multimap. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 650 of file `stl_multimap.h`.

```
5.882.3.21 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >> iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::erase (
const_iterator __first, const_iterator __last ) [inline]
```

Erases a `[first,last)` range of elements from a multimap.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be erased.
<code>__last</code>	Iterator pointing to the end of the range to be erased .

Returns

The iterator `__last`.

This function erases a sequence of elements from a multimap. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 671 of file `stl_multimap.h`.

References `std::multimap<_Key, _Tp, _Compare, _Alloc>::erase()`.

```
5.882.3.22  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp>>> iterator std::multimap<_Key, _Tp, _Compare, _Alloc>::find ( const
            key_type & __x ) [inline]
```

Tries to locate an element in a multimap.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Iterator pointing to sought-after element, or `end()` if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after pair. If unsuccessful it returns the past-the-end (`end()`) iterator.

Definition at line 752 of file `stl_multimap.h`.

```
5.882.3.23  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp>>> template<typename _Kt> auto std::multimap<_Key, _Tp,
            _Compare, _Alloc>::find ( const _Kt & __x )-> decltype(_M.t._M_find_tr(__x)) [inline]
```

Tries to locate an element in a multimap.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Iterator pointing to sought-after element, or `end()` if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after pair. If unsuccessful it returns the past-the-end (`end()`) iterator.

Definition at line 758 of file `stl_multimap.h`.

```
5.882.3.24 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> const_iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::find (
const key_type & __x ) const [inline]
```

Tries to locate an element in a multimap.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Read-only (constant) iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns a constant iterator pointing to the sought after pair. If unsuccessful it returns the past-the-end (`end()`) iterator.

Definition at line 776 of file `stl_multimap.h`.

```
5.882.3.25 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> template<typename _Kt > auto std::multimap< _Key, _Tp,
_Compare, _Alloc >::find ( const _Kt & __x ) const -> decltype(_M_t._M_find_tr(__x)) [inline]
```

Tries to locate an element in a multimap.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Read-only (constant) iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns a constant iterator pointing to the sought after pair. If unsuccessful it returns the past-the-end (`end()`) iterator.

Definition at line 782 of file `stl_multimap.h`.

```
5.882.3.26 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> allocator_type std::multimap< _Key, _Tp, _Compare, _Alloc
>::get_allocator ( ) const [inline], [noexcept]
```

Get a copy of the memory allocation object.

Definition at line 323 of file `stl_multimap.h`.

5.882.3.27 `template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >> iterator std::multimap<_Key, _Tp, _Compare, _Alloc >::insert (
const value_type & __x) [inline]`

Inserts a `std::pair` into the `multimap`.

Parameters

<code>_↔</code>	Pair to be inserted (see std::make_pair for easy creation of pairs).
<code>_x</code>	

Returns

An iterator that points to the inserted (key,value) pair.

This function inserts a (key, value) pair into the multimap. Contrary to a std::map the multimap does not rely on unique keys and thus multiple pairs with the same key can be inserted.

Insertion requires logarithmic time.

Definition at line 520 of file stl_multimap.h.

Referenced by std::multimap< _Key, _Tp, _Compare, _Alloc >::insert().

```
5.882.3.28 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >> iterator std::multimap< _Key, _Tp, _Compare, _Alloc >::insert (
const_iterator __position, const value_type & __x ) [inline]
```

Inserts a std::pair into the multimap.

Parameters

<code>__position</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__x</code>	Pair to be inserted (see std::make_pair for easy creation of pairs).

Returns

An iterator that points to the inserted (key,value) pair.

This function inserts a (key, value) pair into the multimap. Contrary to a std::map the multimap does not rely on unique keys and thus multiple pairs with the same key can be inserted. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.↔html#containers.associative.insert_hints

Insertion requires logarithmic time (if the hint is not taken).

Definition at line 554 of file stl_multimap.h.

References std::multimap< _Key, _Tp, _Compare, _Alloc >::insert().

```
5.882.3.29 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp> >> template<typename _InputIterator > void std::multimap< _Key,
_Tp, _Compare, _Alloc >::insert ( _InputIterator __first, _InputIterator __last ) [inline]
```

A template function that attempts to insert a range of elements.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be inserted.
<code>__last</code>	Iterator pointing to the end of the range.

Complexity similar to that of the range constructor.

Definition at line 581 of file `stl_multimap.h`.

```
5.882.3.30  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp> >> void std::multimap< _Key, _Tp, _Compare, _Alloc >::insert (
            initializer_list< value_type > _I ) [inline]
```

Attempts to insert a list of `std::pairs` into the multimap.

Parameters

<code>↔</code>	A <code>std::initializer_list<value_type></code> of pairs to be inserted.
<code>__↔</code>	
<code>↔</code>	
<code>__↔</code>	
<code>/</code>	

Complexity similar to that of the range constructor.

Definition at line 593 of file `stl_multimap.h`.

References `std::multimap< _Key, _Tp, _Compare, _Alloc >::insert()`.

```
5.882.3.31  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp> >> key_compare std::multimap< _Key, _Tp, _Compare, _Alloc
            >::key_comp ( ) const [inline]
```

Returns the key comparison object out of which the multimap was constructed.

Definition at line 726 of file `stl_multimap.h`.

```
5.882.3.32  template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
            std::allocator<std::pair<const _Key, _Tp> >> iterator std::multimap< _Key, _Tp, _Compare, _Alloc
            >::lower_bound ( const key_type & _x ) [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

<code>__↔</code>	Key of (key, value) pair to be located.
<code>_x</code>	

Returns

Iterator pointing to first element equal to or greater than key, or end().


This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

Definition at line 818 of file stl_multimap.h.

```
5.882.3.33 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> template<typename _Kt > auto std::multimap< _Key, _Tp,
_Compare, _Alloc >::lower_bound ( const _Kt & __x )-> decltype(_M_t._M_lower_bound_tr(__x)) [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

 __x	Key of (key, value) pair to be located.
--	---

Returns

Iterator pointing to first element equal to or greater than key, or end().


This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

Definition at line 824 of file stl_multimap.h.

```
5.882.3.34 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> const_iterator std::multimap< _Key, _Tp, _Compare, _Alloc
>::lower_bound ( const key_type & __x ) const [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

 __x	Key of (key, value) pair to be located.
--	---

Returns

Read-only (constant) iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful the iterator will point to the next greatest element or, if no such greater element exists, to end().

Definition at line 843 of file stl_multimap.h.

```
5.882.3.35 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> template<typename _Kt > auto std::multimap< _Key, _Tp,
_Compare, _Alloc >::lower_bound ( const _Kt & __x ) const -> decltype(_M_t._M_lower_bound_tr(__x)) [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Read-only (constant) iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful the iterator will point to the next greatest element or, if no such greater element exists, to end().

Definition at line 849 of file `stl_multimap.h`.

```
5.882.3.36 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> size_type std::multimap< _Key, _Tp, _Compare, _Alloc >::max_size
( ) const [inline], [noexcept]
```

Returns the maximum size of the multimap.

Definition at line 450 of file `stl_multimap.h`.

```
5.882.3.37 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> multimap& std::multimap< _Key, _Tp, _Compare, _Alloc
>::operator= ( const multimap< _Key, _Tp, _Compare, _Alloc > & __x ) [inline]
```

Multimap assignment operator.

The dtor only erases the elements, and note that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Parameters

<code>__x</code>	A multimap of identical element and allocator types.
------------------	--

All the elements of `__x` are copied, but unlike the copy constructor, the allocator object is not copied.

Definition at line 291 of file `stl_multimap.h`.

```
5.882.3.38 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> multimap& std::multimap<_Key, _Tp, _Compare, _Alloc
>::operator= ( multimap<_Key, _Tp, _Compare, _Alloc> && ) [default]
```

Move assignment operator.

```
5.882.3.39 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> multimap& std::multimap<_Key, _Tp, _Compare, _Alloc
>::operator= ( initializer_list<value_type> &__l ) [inline]
```

Multimap list assignment operator.

Parameters

↩	An initializer_list.
↩	
↩	
↩	
/	

This function fills a multimap with copies of the elements in the initializer list __l.

Note that the assignment completely changes the multimap and that the resulting multimap's size is the same as the number of elements assigned. Old data may be lost.

Definition at line 314 of file stl_multimap.h.

```
5.882.3.40 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> reverse_iterator std::multimap<_Key, _Tp, _Compare, _Alloc
>::rbegin ( ) [inline], [noexcept]
```

Returns a read/write reverse iterator that points to the last pair in the multimap. Iteration is done in descending order according to the keys.

Definition at line 369 of file stl_multimap.h.

```
5.882.3.41 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> const_reverse_iterator std::multimap<_Key, _Tp, _Compare,
_Alloc>::rbegin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last pair in the multimap. Iteration is done in descending order according to the keys.

Definition at line 378 of file stl_multimap.h.

```
5.882.3.42 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> reverse_iterator std::multimap<_Key, _Tp, _Compare, _Alloc
>::rend ( ) [inline], [noexcept]
```

Returns a read/write reverse iterator that points to one before the first pair in the multimap. Iteration is done in descending order according to the keys.

Definition at line 387 of file stl_multimap.h.

```
5.882.3.43 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> const_reverse_iterator std::multimap<_Key, _Tp, _Compare,
_Alloc>::rend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to one before the first pair in the multimap. Iteration is done in descending order according to the keys.

Definition at line 396 of file stl_multimap.h.

```
5.882.3.44 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> size_type std::multimap<_Key, _Tp, _Compare, _Alloc>::size ( )
const [inline], [noexcept]
```

Returns the size of the multimap.

Definition at line 445 of file stl_multimap.h.

```
5.882.3.45 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> void std::multimap<_Key, _Tp, _Compare, _Alloc>::swap (
multimap<_Key, _Tp, _Compare, _Alloc> &__x ) [inline], [noexcept]
```

Swaps data with another multimap.

Parameters

<code>__x</code>	A multimap of the same element and allocator types.
------------------	---

This exchanges the elements between two multimaps in constant time. (It is only swapping a pointer, an integer, and an instance of the `Compare` type (which itself is often stateless and empty), so it should be quite fast.) Note that the global `std::swap()` function is specialized such that `std::swap(m1,m2)` will feed to this function.

Definition at line 706 of file stl_multimap.h.

```
5.882.3.46 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> iterator std::multimap<_Key, _Tp, _Compare, _Alloc>
::upper_bound ( const key_type &__x ) [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Iterator pointing to the first element greater than key, or `end()`.

Definition at line 863 of file stl_multimap.h.

```
5.882.3.47 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>>> template<typename _Kt > auto std::multimap< _Key, _Tp,
_Compare, _Alloc >::upper_bound ( const _Kt & __x )-> decltype(_M.t._M_upper_bound_tr(__x)) [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Iterator pointing to the first element greater than key, or end().

Definition at line 869 of file stl_multimap.h.

```
5.882.3.48 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>>> const_iterator std::multimap< _Key, _Tp, _Compare, _Alloc
>::upper_bound ( const key_type & __x ) const [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Read-only (constant) iterator pointing to first iterator greater than key, or end().

Definition at line 883 of file stl_multimap.h.

```
5.882.3.49 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>>> template<typename _Kt > auto std::multimap< _Key,
_Tp, _Compare, _Alloc >::upper_bound ( const _Kt & __x ) const -> decltype(_M.t._M_upper_bound_tr(__x))
[inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>__x</code>	Key of (key, value) pair to be located.
------------------	---

Returns

Read-only (constant) iterator pointing to first iterator greater than key, or end().

Definition at line 889 of file stl_multimap.h.

```
5.882.3.50 template<typename _Key, typename _Tp, typename _Compare = std::less<_Key>, typename _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> value_compare std::multimap<_Key, _Tp, _Compare, _Alloc>
>::value_comp( ) const [inline]
```

Returns a value comparison object, built from the key comparison object out of which the multimap was constructed.

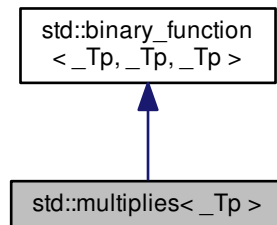
Definition at line 734 of file stl_multimap.h.

The documentation for this class was generated from the following file:

- [stl_multimap.h](#)

5.883 std::multiplies<_Tp> Struct Template Reference

Inheritance diagram for std::multiplies<_Tp>:

**Public Types**

- typedef _Tp [first_argument_type](#)
- typedef _Tp [result_type](#)
- typedef _Tp [second_argument_type](#)

Public Member Functions

- `_GLIBCXX14_CONSTEXPR _Tp operator() (const _Tp &__x, const _Tp &__y) const`

5.883.1 Detailed Description

```
template<typename _Tp>
struct std::multiplies< _Tp >
```

One of the [math functors](#).

Definition at line 153 of file `stl_function.h`.

5.883.2 Member Typedef Documentation

5.883.2.1 `typedef _Tp std::binary_function< _Tp, _Tp, _Tp >::first_argument_type` [inherited]

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.883.2.2 `typedef _Tp std::binary_function< _Tp, _Tp, _Tp >::result_type` [inherited]

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.883.2.3 `typedef _Tp std::binary_function< _Tp, _Tp, _Tp >::second_argument_type` [inherited]

`second_argument_type` is the type of the second argument

Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.884 `std::multiplies< void >` Struct Template Reference

Public Types

- `typedef __is_transparent is_transparent`

Public Member Functions

- `template<typename _Tp, typename _Up > _GLIBCXX14_CONSTEXPR auto operator() (_Tp &&__t, _Up &&__u) const noexcept(noexcept(std::forward< _Tp >\(__t\)*std::forward< _Up >\(__u\)\))) -> decltype(std::forward< _Tp >\(__t\)*std::forward< _Up >\(__u\))`

5.884.1 Detailed Description

```
template<>
struct std::multiplies< void >
```

One of the [math functors](#).

Definition at line 260 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.885 `std::multiset< _Key, _Compare, _Alloc >` Class Template Reference

Public Types

- typedef `_Alloc` **allocator_type**
- typedef `_Rep_type::const_iterator` **const_iterator**
- typedef `_Alloc_traits::const_pointer` **const_pointer**
- typedef `_Alloc_traits::const_reference` **const_reference**
- typedef `_Rep_type::const_reverse_iterator` **const_reverse_iterator**
- typedef `_Rep_type::difference_type` **difference_type**
- typedef `_Rep_type::const_iterator` **iterator**
- typedef `_Compare` **key_compare**
- typedef `_Key` **key_type**
- typedef `_Alloc_traits::pointer` **pointer**
- typedef `_Alloc_traits::reference` **reference**
- typedef `_Rep_type::const_reverse_iterator` **reverse_iterator**
- typedef `_Rep_type::size_type` **size_type**
- typedef `_Compare` **value_compare**
- typedef `_Key` **value_type**

Public Member Functions

- [multiset](#) () noexcept(*/*conditional */*)
- [multiset](#) (const `_Compare` &__comp, const `allocator_type` &__a=allocator_type())
- template<typename `_InputIterator` >
[multiset](#) (`_InputIterator` __first, `_InputIterator` __last)
- template<typename `_InputIterator` >
[multiset](#) (`_InputIterator` __first, `_InputIterator` __last, const `_Compare` &__comp, const `allocator_type` &__a=allocator_type())
- [multiset](#) (const [multiset](#) &__x)
- [multiset](#) ([multiset](#) &&__x) noexcept(is_nothrow_copy_constructible< `_Compare` >::value)
- [multiset](#) ([initializer_list](#)< `value_type` > __l, const `_Compare` &__comp=_Compare(), const `allocator_type` &__a=allocator_type())
- [multiset](#) (const `allocator_type` &__a)
- [multiset](#) (const [multiset](#) &__m, const `allocator_type` &__a)

- `multiset` (`multiset` && __m, const allocator_type & __a) noexcept(is_nothrow_copy_constructible< _Compare > &::value && _Alloc_traits::S_always_equal())
 - `multiset` (`initializer_list`< value_type > __l, const allocator_type & __a)
 - template<typename _InputIterator >
`multiset` (_InputIterator __first, _InputIterator __last, const allocator_type & __a)
 - iterator `begin` () const noexcept
 - iterator `cbegin` () const noexcept
 - iterator `cend` () const noexcept
 - void `clear` () noexcept
 - `reverse_iterator` `crbegin` () const noexcept
 - `reverse_iterator` `crend` () const noexcept
 - template<typename... _Args>
iterator `emplace` (_Args &&... __args)
 - template<typename... _Args>
iterator `emplace_hint` (const_iterator __pos, _Args &&... __args)
 - bool `empty` () const noexcept
 - iterator `end` () const noexcept
 - _GLIBCXX_ABI_TAG_CXX11 iterator `erase` (const_iterator __position)
 - size_type `erase` (const key_type & __x)
 - _GLIBCXX_ABI_TAG_CXX11 iterator `erase` (const_iterator __first, const_iterator __last)
 - allocator_type `get_allocator` () const noexcept
 - iterator `insert` (const value_type & __x)
 - iterator `insert` (value_type && __x)
 - iterator `insert` (const_iterator __position, const value_type & __x)
 - iterator `insert` (const_iterator __position, value_type && __x)
 - template<typename _InputIterator >
void `insert` (_InputIterator __first, _InputIterator __last)
 - void `insert` (`initializer_list`< value_type > __l)
 - key_compare `key_comp` () const
 - size_type `max_size` () const noexcept
 - `multiset` & `operator=` (const `multiset` & __x)
 - `multiset` & `operator=` (`multiset` &&) = default
 - `multiset` & `operator=` (`initializer_list`< value_type > __l)
 - `reverse_iterator` `rbegin` () const noexcept
 - `reverse_iterator` `rend` () const noexcept
 - size_type `size` () const noexcept
 - void `swap` (`multiset` & __x) noexcept(*conditional *)
 - value_compare `value_comp` () const
-
- size_type `count` (const key_type & __x) const
 - template<typename _Kt >
auto `count` (const _Kt & __x) const -> decltype(_M_t._M_count_tr(__x))
-
- iterator `find` (const key_type & __x)
 - const_iterator `find` (const key_type & __x) const
 - template<typename _Kt >
auto `find` (const _Kt & __x) -> decltype(iterator
 - template<typename _Kt >
auto `find` (const _Kt & __x) const -> decltype(const_iterator

- iterator [lower_bound](#) (const key_type &__x)
- const_iterator [lower_bound](#) (const key_type &__x) const
- template<typename _Kt >
auto [lower_bound](#) (const _Kt &__x) -> decltype(_M_t._M_lower_bound_tr(__x))
- template<typename _Kt >
auto [lower_bound](#) (const _Kt &__x) const -> decltype(_M_t._M_lower_bound_tr(__x))

- iterator [upper_bound](#) (const key_type &__x)
- const_iterator [upper_bound](#) (const key_type &__x) const
- template<typename _Kt >
auto [upper_bound](#) (const _Kt &__x) -> decltype(_M_t._M_upper_bound_tr(__x))
- template<typename _Kt >
auto [upper_bound](#) (const _Kt &__x) const -> decltype(_M_t._M_upper_bound_tr(__x))

- [std::pair](#)< iterator, iterator > [equal_range](#) (const key_type &__x)
- [std::pair](#)< const_iterator, const_iterator > [equal_range](#) (const key_type &__x) const
- template<typename _Kt >
auto [equal_range](#) (const _Kt &__x) -> decltype(_M_t._M_equal_range_tr(__x))
- template<typename _Kt >
auto [equal_range](#) (const _Kt &__x) const -> decltype(_M_t._M_equal_range_tr(__x))

Friends

- template<typename _K1, typename _C1, typename _A1 >
bool **operator**< (const [multiset](#)< _K1, _C1, _A1 > &, const [multiset](#)< _K1, _C1, _A1 > &)
- template<typename _K1, typename _C1, typename _A1 >
bool **operator**== (const [multiset](#)< _K1, _C1, _A1 > &, const [multiset](#)< _K1, _C1, _A1 > &)

5.885.1 Detailed Description

```
template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
class std::multiset< _Key, _Compare, _Alloc >
```

A standard container made up of elements, which can be retrieved in logarithmic time.

Template Parameters

<code>_Key</code>	Type of key objects.
<code>_Compare</code>	Comparison function object type, defaults to <code>less<_Key></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_Key></code> .

Meets the requirements of a [container](#), a [reversible container](#), and an [associative container](#) (using equivalent keys). For a `multiset<Key>` the `key_type` and `value_type` are `Key`.

Multisets support bidirectional iterators.

The private tree data is declared exactly the same way for set and multiset; the distinction is made entirely in how the tree functions are called (*_unique versus *_equal, same as the standard).

Definition at line 92 of file stl_multiset.h.

5.885.2 Constructor & Destructor Documentation

5.885.2.1 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::multiset<_Key, _Compare, _Alloc>::multiset () [inline], [noexcept]`

Default constructor creates no elements.

Definition at line 140 of file stl_multiset.h.

5.885.2.2 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::multiset<_Key, _Compare, _Alloc>::multiset (const _Compare & __comp, const allocator_type & __a =
allocator_type()) [inline], [explicit]`

Creates a multiset with no elements.

Parameters

<code>__comp</code>	Comparator to use.
<code>__a</code>	An allocator object.

Definition at line 152 of file stl_multiset.h.

5.885.2.3 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename _InputIterator > std::multiset<_Key, _Compare, _Alloc>::multiset (_InputIterator __first,
_InputIterator __last) [inline]`

Builds a multiset from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Create a multiset consisting of copies of the elements from [first,last). This is linear in N if the range is already sorted, and NlogN otherwise (where N is distance(__first,__last)).

Definition at line 166 of file stl_multiset.h.

5.885.2.4 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename _InputIterator > std::multiset<_Key, _Compare, _Alloc>::multiset (_InputIterator __first,
_InputIterator __last, const _Compare & __comp, const allocator_type & __a = allocator_type())
[inline]`

Builds a multiset from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__comp</code>	A comparison functor.
<code>__a</code>	An allocator object.

Create a multiset consisting of copies of the elements from `[__first,__last)`. This is linear in N if the range is already sorted, and NlogN otherwise (where N is distance(`__first,__last`)).

Definition at line 182 of file `stl_multiset.h`.

```
5.885.2.5  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            std::multiset< _Key, _Compare, _Alloc >::multiset ( const multiset< _Key, _Compare, _Alloc > & __x )
            [inline]
```

Multiset copy constructor.

Parameters

<code>__x</code>	A multiset of identical element and allocator types.
------------------	--

The newly-created multiset uses a copy of the allocation object used by `__x`.

Definition at line 195 of file `stl_multiset.h`.

```
5.885.2.6  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            std::multiset< _Key, _Compare, _Alloc >::multiset ( multiset< _Key, _Compare, _Alloc > && __x ) [inline],
            [noexcept]
```

Multiset move constructor.

Parameters

<code>__x</code>	A multiset of identical element and allocator types.
------------------	--

The newly-created multiset contains the exact contents of `__x`. The contents of `__x` are a valid, but unspecified multiset.

Definition at line 206 of file `stl_multiset.h`.

```
5.885.2.7  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            std::multiset< _Key, _Compare, _Alloc >::multiset ( initializer_list< value_type > __l, const _Compare & __comp
            = _Compare(), const allocator_type & __a = allocator_type() ) [inline]
```

Builds a multiset from an `initializer_list`.

Parameters

<code>__l</code>	An initializer_list.
<code>__comp</code>	A comparison functor.
<code>__a</code>	An allocator object.

Create a multiset consisting of copies of the elements from the list. This is linear in N if the list is already sorted, and NlogN otherwise (where N is `__l.size()`).

Definition at line 220 of file `stl_multiset.h`.

```
5.885.2.8  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
           std::multiset< _Key, _Compare, _Alloc >::multiset ( const allocator_type & __a )  [inline], [explicit]
```

Allocator-extended default constructor.

Definition at line 228 of file `stl_multiset.h`.

```
5.885.2.9  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
           std::multiset< _Key, _Compare, _Alloc >::multiset ( const multiset< _Key, _Compare, _Alloc > & __m, const
           allocator_type & __a )  [inline]
```

Allocator-extended copy constructor.

Definition at line 232 of file `stl_multiset.h`.

```
5.885.2.10 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
           std::multiset< _Key, _Compare, _Alloc >::multiset ( multiset< _Key, _Compare, _Alloc > && __m, const
           allocator_type & __a )  [inline], [noexcept]
```

Allocator-extended move constructor.

Definition at line 236 of file `stl_multiset.h`.

```
5.885.2.11 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
           std::multiset< _Key, _Compare, _Alloc >::multiset ( initializer_list< value_type > __l, const allocator_type &
           __a )  [inline]
```

Allocator-extended initializer-list constructor.

Definition at line 242 of file `stl_multiset.h`.

```
5.885.2.12 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
           template<typename _InputIterator > std::multiset< _Key, _Compare, _Alloc >::multiset ( _InputIterator __first,
           _InputIterator __last, const allocator_type & __a )  [inline]
```

Allocator-extended range constructor.

Definition at line 248 of file `stl_multiset.h`.

5.885.3 Member Function Documentation

5.885.3.1 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> iterator
std::multiset<_Key, _Compare, _Alloc>::begin () const [inline], [noexcept]`

Returns a read-only (constant) iterator that points to the first element in the multiset. Iteration is done in ascending order according to the keys.

Definition at line 313 of file `stl_multiset.h`.

5.885.3.2 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> iterator
std::multiset<_Key, _Compare, _Alloc>::cbegin () const [inline], [noexcept]`

Returns a read-only (constant) iterator that points to the first element in the multiset. Iteration is done in ascending order according to the keys.

Definition at line 350 of file `stl_multiset.h`.

5.885.3.3 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> iterator
std::multiset<_Key, _Compare, _Alloc>::cend () const [inline], [noexcept]`

Returns a read-only (constant) iterator that points one past the last element in the multiset. Iteration is done in ascending order according to the keys.

Definition at line 359 of file `stl_multiset.h`.

5.885.3.4 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> void
std::multiset<_Key, _Compare, _Alloc>::clear () [inline], [noexcept]`

Erases all elements in a multiset. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 635 of file `stl_multiset.h`.

5.885.3.5 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
size_type std::multiset<_Key, _Compare, _Alloc>::count (const key_type &__x) const [inline]`

Finds the number of elements with given key.

Parameters

<code>__x</code>	Key of elements to be located.
------------------	--------------------------------

Returns

Number of elements with specified key.

Definition at line 647 of file `stl_multiset.h`.

```
5.885.3.6 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename _Kt > auto std::multiset< _Key, _Compare, _Alloc >::count ( const _Kt & __x ) const ->
decltype(_M_t._M_count_tr(__x)) [inline]
```

Finds the number of elements with given key.

Parameters

<code>__x</code>	Key of elements to be located.
------------------	--------------------------------

Returns

Number of elements with specified key.

Definition at line 653 of file stl_multiset.h.

```
5.885.3.7 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
reverse_iterator std::multiset< _Key, _Compare, _Alloc >::crbegin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last element in the multiset. Iteration is done in descending order according to the keys.

Definition at line 368 of file stl_multiset.h.

```
5.885.3.8 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
reverse_iterator std::multiset< _Key, _Compare, _Alloc >::crend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last element in the multiset. Iteration is done in descending order according to the keys.

Definition at line 377 of file stl_multiset.h.

```
5.885.3.9 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename... _Args> iterator std::multiset< _Key, _Compare, _Alloc >::emplace ( _Args &&... __args )
[inline]
```

Builds and inserts an element into the multiset.

Parameters

<code>__args</code>	Arguments used to generate the element instance to be inserted.
---------------------	---

Returns

An iterator that points to the inserted element.

This function inserts an element into the multiset. Contrary to a `std::set` the multiset does not rely on unique keys and thus multiple copies of the same element can be inserted.

Insertion requires logarithmic time.

Definition at line 428 of file `stl_multiset.h`.

```
5.885.3.10  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            template<typename... _Args> iterator std::multiset<_Key, _Compare, _Alloc>::emplace_hint ( const_iterator
            __pos, _Args &&... __args ) [inline]
```

Builds and inserts an element into the multiset.

Parameters

<code>__pos</code>	An iterator that serves as a hint as to where the element should be inserted.
<code>__args</code>	Arguments used to generate the element instance to be inserted.

Returns

An iterator that points to the inserted element.

This function inserts an element into the multiset. Contrary to a `std::set` the multiset does not rely on unique keys and thus multiple copies of the same element can be inserted.

Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints for more on *hinting*.

Insertion requires logarithmic time (if the hint is not taken).

Definition at line 454 of file `stl_multiset.h`.

```
5.885.3.11  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> bool
            std::multiset<_Key, _Compare, _Alloc>::empty ( ) const [inline], [noexcept]
```

Returns true if the set is empty.

Definition at line 383 of file `stl_multiset.h`.

```
5.885.3.12  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> iterator
            std::multiset<_Key, _Compare, _Alloc>::end ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the multiset. Iteration is done in ascending order according to the keys.

Definition at line 322 of file `stl_multiset.h`.

```
5.885.3.13  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            std::pair<iterator, iterator> std::multiset<_Key, _Compare, _Alloc>::equal_range ( const key_type & __x )
            [inline]
```

Finds a subsequence matching given key.

Parameters

$_↔$	Key to be located.
$_X$	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
               c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multisets.

Definition at line 777 of file stl_multiset.h.

```
5.885.3.14 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            std::pair<const_iterator, const_iterator> std::multiset<_Key, _Compare, _Alloc >::equal_range ( const key_type
            & __x ) const    [inline]
```

Finds a subsequence matching given key.

Parameters

$_↔$	Key to be located.
$_X$	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
               c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multisets.

Definition at line 781 of file stl_multiset.h.

```
5.885.3.15 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            template<typename _Kt > auto std::multiset<_Key, _Compare, _Alloc >::equal_range ( const _Kt & __x ) ->
            decltype(_M.t._M_equal_range_tr(__x))    [inline]
```

Finds a subsequence matching given key.

Parameters

<code>_↔</code>	Key to be located.
<code>_X</code>	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
               c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multisets.

Definition at line 787 of file stl_multiset.h.

```
5.885.3.16 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            template<typename _Kt > auto std::multiset<_Key, _Compare, _Alloc >::equal_range ( const _Kt & __x ) const ->
            decltype(_M_t._M_equal_range_tr(__x)) [inline]
```

Finds a subsequence matching given key.

Parameters

<code>_↔</code>	Key to be located.
<code>_X</code>	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
               c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multisets.

Definition at line 793 of file stl_multiset.h.

```
5.885.3.17 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            _GLIBCXX_ABI_TAG_CXX11 iterator std::multiset<_Key, _Compare, _Alloc >::erase ( const_iterator __position )
            [inline]
```

Erases an element from a multiset.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following *position* prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from a multiset. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 556 of file `stl_multiset.h`.

Referenced by `std::multiset< _Key, _Compare, _Alloc >::erase()`.

```
5.885.3.18  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            size_type std::multiset< _Key, _Compare, _Alloc >::erase ( const key_type & __x )  [inline]
```

Erases elements according to the provided key.

Parameters

<code>__x</code>	Key of element to be erased.
------------------	------------------------------

Returns

The number of elements erased.

This function erases all elements located by the given key from a multiset. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 586 of file `stl_multiset.h`.

```
5.885.3.19  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            _GLIBCXX_ABI_TAG_CXX11 iterator std::multiset< _Key, _Compare, _Alloc >::erase ( const_iterator __first,
            const_iterator __last )  [inline]
```

Erases a `[first,last)` range of elements from a multiset.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be erased.
<code>__last</code>	Iterator pointing to the end of the range to be erased.

Returns

The iterator *last*.

This function erases a sequence of elements from a multiset. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 608 of file `stl_multiset.h`.

References `std::multiset<_Key, _Compare, _Alloc >::erase()`.

5.885.3.20 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> iterator
std::multiset<_Key, _Compare, _Alloc >::find (const key_type & __x) [inline]`

Tries to locate an element in a set.

Parameters

<code>__x</code>	Element to be located.
------------------	------------------------

Returns

Iterator pointing to sought-after element, or `end()` if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (`end()`) iterator.

Definition at line 673 of file `stl_multiset.h`.

5.885.3.21 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
const_iterator std::multiset<_Key, _Compare, _Alloc >::find (const key_type & __x) const [inline]`

Tries to locate an element in a set.

Parameters

<code>__x</code>	Element to be located.
------------------	------------------------

Returns

Iterator pointing to sought-after element, or `end()` if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (`end()`) iterator.

Definition at line 677 of file `stl_multiset.h`.

```
5.885.3.22 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename _Kt > auto std::multiset< _Key, _Compare, _Alloc >::find ( const _Kt & __x ) ->
decltype(iterator) [inline]
```

Tries to locate an element in a set.

Parameters

<code>__x</code>	Element to be located.
------------------	------------------------

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (end()) iterator.

Definition at line 683 of file stl_multiset.h.

```
5.885.3.23 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename _Kt > auto std::multiset< _Key, _Compare, _Alloc >::find ( const _Kt & __x ) const ->
decltype(const_iterator) [inline]
```

Tries to locate an element in a set.

Parameters

<code>__x</code>	Element to be located.
------------------	------------------------

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (end()) iterator.

Definition at line 689 of file stl_multiset.h.

```
5.885.3.24 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
allocator_type std::multiset< _Key, _Compare, _Alloc >::get_allocator ( ) const [inline], [noexcept]
```

Returns the memory allocation object.

Definition at line 304 of file stl_multiset.h.

```
5.885.3.25 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> iterator
std::multiset< _Key, _Compare, _Alloc >::insert ( const value_type & __x ) [inline]
```

Inserts an element into the multiset.

Parameters

<code>_↔</code>	Element to be inserted.
<code>_x</code>	

Returns

An iterator that points to the inserted element.

This function inserts an element into the multiset. Contrary to a `std::set` the multiset does not rely on unique keys and thus multiple copies of the same element can be inserted.

Insertion requires logarithmic time.

Definition at line 473 of file `stl_multiset.h`.

Referenced by `std::multiset<_Key, _Compare, _Alloc >::insert()`.

```
5.885.3.26  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> iterator
            std::multiset<_Key, _Compare, _Alloc >::insert ( const_iterator __position, const value_type & __x ) [inline]
```

Inserts an element into the multiset.

Parameters

<code>__position</code>	An iterator that serves as a hint as to where the element should be inserted.
<code>__x</code>	Element to be inserted.

Returns

An iterator that points to the inserted element.

This function inserts an element into the multiset. Contrary to a `std::set` the multiset does not rely on unique keys and thus multiple copies of the same element can be inserted.

Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.↔associative.insert_hints for more on *hinting*.

Insertion requires logarithmic time (if the hint is not taken).

Definition at line 503 of file `stl_multiset.h`.

References `std::multiset<_Key, _Compare, _Alloc >::insert()`.

```
5.885.3.27  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            template<typename _InputIterator > void std::multiset<_Key, _Compare, _Alloc >::insert ( _InputIterator __first,
            _InputIterator __last ) [inline]
```

A template function that tries to insert a range of elements.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be inserted.
<code>__last</code>	Iterator pointing to the end of the range.

Complexity similar to that of the range constructor.

Definition at line 522 of file `stl_multiset.h`.

5.885.3.28 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> void std::multiset<_Key, _Compare, _Alloc>::insert(initializer_list<value_type> __l) [inline]`

Attempts to insert a list of elements into the multiset.

Parameters

<code>↔</code>	A <code>std::initializer_list<value_type></code> of elements to be inserted.
<code>__↔</code>	
<code>↔</code>	
<code>__↔</code>	
<code>/</code>	

Complexity similar to that of the range constructor.

Definition at line 534 of file `stl_multiset.h`.

References `std::multiset<_Key, _Compare, _Alloc>::insert()`.

5.885.3.29 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> key_compare std::multiset<_Key, _Compare, _Alloc>::key_comp() const [inline]`

Returns the comparison object.

Definition at line 296 of file `stl_multiset.h`.

5.885.3.30 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> iterator std::multiset<_Key, _Compare, _Alloc>::lower_bound(const key_type & __x) [inline]`

Finds the beginning of a subsequence matching given key.

Parameters

<code>__↔</code>	Key to be located.
<code>__x</code>	

Returns

Iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

Definition at line 708 of file stl_multiset.h.

```
5.885.3.31  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            const_iterator std::multiset<_Key, _Compare, _Alloc>::lower_bound ( const key_type & __x ) const    [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

\leftarrow	Key to be located.
$_x$	

Returns

Iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

Definition at line 712 of file stl_multiset.h.

```
5.885.3.32  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            template<typename _Kt > auto std::multiset<_Key, _Compare, _Alloc>::lower_bound ( const _Kt & __x ) ->
            decltype(_M_t._M_lower_bound_tr(__x))    [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

\leftarrow	Key to be located.
$_x$	

Returns

Iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

Definition at line 718 of file stl_multiset.h.

```
5.885.3.33 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename _Kt > auto std::multiset<_Key, _Compare, _Alloc >::lower_bound ( const _Kt & __x ) const ->
decltype(_M_t._M_lower_bound_tr(__x)) [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

<code>__x</code>	Key to be located.
------------------	--------------------

Returns

Iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

Definition at line 724 of file stl_multiset.h.

```
5.885.3.34 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
size_type std::multiset<_Key, _Compare, _Alloc >::max_size ( ) const [inline], [noexcept]
```

Returns the maximum size of the set.

Definition at line 393 of file stl_multiset.h.

```
5.885.3.35 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
multiset& std::multiset<_Key, _Compare, _Alloc >::operator= ( const multiset<_Key, _Compare, _Alloc > & __x
) [inline]
```

Multiset assignment operator.

Parameters

<code>__x</code>	A multiset of identical element and allocator types.
------------------	--

All the elements of `__x` are copied, but unlike the copy constructor, the allocator object is not copied.

Definition at line 262 of file stl_multiset.h.

```
5.885.3.36 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
multiset& std::multiset<_Key, _Compare, _Alloc >::operator= ( multiset<_Key, _Compare, _Alloc > && )
[default]
```

Move assignment operator.

```
5.885.3.37  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            multiset& std::multiset< _Key, _Compare, _Alloc >::operator= ( initializer_list< value_type > __l )
            [inline]
```

Multiset list assignment operator.

Parameters

↩	An initializer_list.
__↩	
↩	
__↩	
l	

This function fills a multiset with copies of the elements in the initializer list `__l`.

Note that the assignment completely changes the multiset and that the resulting multiset's size is the same as the number of elements assigned. Old data may be lost.

Definition at line 285 of file `stl_multiset.h`.

```
5.885.3.38  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            reverse_iterator std::multiset< _Key, _Compare, _Alloc >::rbegin ( ) const  [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last element in the multiset. Iteration is done in descending order according to the keys.

Definition at line 331 of file `stl_multiset.h`.

```
5.885.3.39  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            reverse_iterator std::multiset< _Key, _Compare, _Alloc >::rend ( ) const  [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last element in the multiset. Iteration is done in descending order according to the keys.

Definition at line 340 of file `stl_multiset.h`.

```
5.885.3.40  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            size_type std::multiset< _Key, _Compare, _Alloc >::size ( ) const  [inline], [noexcept]
```

Returns the size of the set.

Definition at line 388 of file `stl_multiset.h`.

```
5.885.3.41  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> void
            std::multiset< _Key, _Compare, _Alloc >::swap ( multiset< _Key, _Compare, _Alloc > & __x ) [inline],
            [noexcept]
```

Swaps data with another multiset.

Parameters

<code>__x</code>	A multiset of the same element and allocator types.
-------------------------	---

This exchanges the elements between two multisets in constant time. (It is only swapping a pointer, an integer, and an instance of the `Compare` type (which itself is often stateless and empty), so it should be quite fast.) Note that the global `std::swap()` function is specialized such that `std::swap(s1,s2)` will feed to this function.

Definition at line 408 of file `stl_multiset.h`.

```
5.885.3.42  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> iterator
            std::multiset< _Key, _Compare, _Alloc >::upper_bound ( const key_type & __x )  [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>__x</code>	Key to be located.
-------------------------	--------------------

Returns

Iterator pointing to the first element greater than key, or `end()`.

Definition at line 738 of file `stl_multiset.h`.

```
5.885.3.43  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            const_iterator std::multiset< _Key, _Compare, _Alloc >::upper_bound ( const key_type & __x ) const  [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>__x</code>	Key to be located.
-------------------------	--------------------

Returns

Iterator pointing to the first element greater than key, or `end()`.

Definition at line 742 of file `stl_multiset.h`.

```
5.885.3.44  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            template<typename _Kt > auto std::multiset< _Key, _Compare, _Alloc >::upper_bound ( const _Kt & __x ) ->
            decltype(_M_t._M_upper_bound_tr(__x))  [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>_↔</code>	Key to be located.
<code>_x</code>	

Returns

Iterator pointing to the first element greater than key, or end().

Definition at line 748 of file `stl_multiset.h`.

```
5.885.3.45  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            template<typename _Kt > auto std::multiset<_Key, _Compare, _Alloc>::upper_bound ( const _Kt & __x ) const ->
            decltype(_M_t._M_upper_bound_tr(__x))    [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>_↔</code>	Key to be located.
<code>_x</code>	

Returns

Iterator pointing to the first element greater than key, or end().

Definition at line 754 of file `stl_multiset.h`.

```
5.885.3.46  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            value_compare std::multiset<_Key, _Compare, _Alloc>::value_comp ( ) const    [inline]
```

Returns the comparison object.

Definition at line 300 of file `stl_multiset.h`.

The documentation for this class was generated from the following file:

- [stl_multiset.h](#)

5.886 std::mutex Class Reference

Inherits `std::__mutex_base`.

Public Types

- typedef `__native_type *` **native_handle_type**

Public Member Functions

- **mutex** (const `mutex` &)=delete
- void **lock** ()
- native_handle_type **native_handle** ()
- `mutex` & **operator=** (const `mutex` &)=delete
- bool **try_lock** () noexcept
- void **unlock** ()

Private Types

- typedef `__gthread_mutex_t` **__native_type**

Private Attributes

- `__native_type` **_M_mutex**

5.886.1 Detailed Description

The standard mutex type.

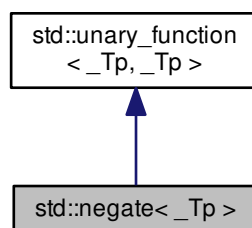
Definition at line 86 of file `std_mutex.h`.

The documentation for this class was generated from the following file:

- [std_mutex.h](#)

5.887 `std::negate<_Tp>` Struct Template Reference

Inheritance diagram for `std::negate<_Tp>`:



Public Types

- typedef `_Tp` [argument_type](#)
- typedef `_Tp` [result_type](#)

Public Member Functions

- `_GLIBCXX14_CONSTEXPR _Tp` **operator()** (const `_Tp` &__x) const

5.887.1 Detailed Description

```
template<typename _Tp>
struct std::negate<_Tp>
```

One of the [math functors](#).

Definition at line 162 of file `stl_function.h`.

5.887.2 Member Typedef Documentation

5.887.2.1 typedef `_Tp` `std::unary_function<_Tp, _Tp>::argument_type` [inherited]

`argument_type` is the type of the argument

Definition at line 108 of file `stl_function.h`.

5.887.2.2 typedef `_Tp` `std::unary_function<_Tp, _Tp>::result_type` [inherited]

`result_type` is the return type

Definition at line 111 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.888 `std::negate< void >` Struct Template Reference

Public Types

- typedef `__is_transparent` **is_transparent**

Public Member Functions

- `template<typename _Tp>`
`_GLIBCXX14_CONSTEXPR auto operator() (_Tp &&__t) const noexcept(noexcept(-std::forward<_Tp>(__t)))`
`-> decltype(-std::forward<_Tp>(__t))`

5.888.1 Detailed Description

```
template<>
struct std::negate< void >
```

One of the [math functors](#).

Definition at line 305 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.889 `std::negative_binomial_distribution<_IntType>` Class Template Reference

Classes

- struct [param_type](#)

Public Types

- typedef `_IntType` [result_type](#)

Public Member Functions

- **`negative_binomial_distribution`** (`_IntType` __k=1, `double` __p=0.5)
- **`negative_binomial_distribution`** (const [param_type](#) &__p)
- `template<typename _ForwardIterator, typename _UniformRandomNumberGenerator>`
`void __generate (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)`
- `template<typename _ForwardIterator, typename _UniformRandomNumberGenerator>`
`void __generate (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const`
`param_type &__p)`
- `template<typename _UniformRandomNumberGenerator>`
`void __generate (result_type *__f, result_type *__t, _UniformRandomNumberGenerator &__urng)`
- `template<typename _UniformRandomNumberGenerator>`
`void __generate (result_type *__f, result_type *__t, _UniformRandomNumberGenerator &__urng, const`
`param_type &__p)`
- `_IntType k () const`
- `result_type max () const`
- `result_type min () const`
- `template<typename _UniformRandomNumberGenerator>`
`result_type operator() (_UniformRandomNumberGenerator &__urng)`
- `template<typename _UniformRandomNumberGenerator>`
`result_type operator() (_UniformRandomNumberGenerator &__urng, const param_type &__p)`
- `double p () const`
- `param_type param () const`
- `void param (const param_type &__param)`
- `void reset ()`

Friends

- `template<typename _IntType1, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::negative_binomial_distribution< _IntType1 > &__x)`
- `bool operator== (const negative_binomial_distribution &__d1, const negative_binomial_distribution &__d2)`
- `template<typename _IntType1, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & operator>> (std::basic_istream< _CharT, _Traits > &__is, std::`
`::negative_binomial_distribution< _IntType1 > &__x)`

5.889.1 Detailed Description

```
template<typename _IntType = int>
class std::negative_binomial_distribution< _IntType >
```

A negative_binomial_distribution random number distribution.

The formula for the negative binomial probability mass function is $p(i) = \binom{n}{i} p^i (1-p)^{t-i}$ where t and p are the parameters of the distribution.

Definition at line 4043 of file random.h.

5.889.2 Member Typedef Documentation

5.889.2.1 `template<typename _IntType = int> typedef _IntType std::negative_binomial_distribution< _IntType`
`>::result_type`

The type of the range of the distribution.

Definition at line 4046 of file random.h.

5.889.3 Member Function Documentation

5.889.3.1 `template<typename _IntType = int> _IntType std::negative_binomial_distribution< _IntType >::k () const`
`[inline]`

Return the k parameter of the distribution.

Definition at line 4101 of file random.h.

5.889.3.2 `template<typename _IntType = int> result_type std::negative_binomial_distribution< _IntType >::max ()`
`const [inline]`

Returns the least upper bound value of the distribution.

Definition at line 4137 of file random.h.

References `std::numeric_limits< _Tp >::max()`.

```
5.889.3.3 template<typename _IntType = int> result_type std::negative_binomial_distribution< _IntType >::min ( )
const [inline]
```

Returns the greatest lower bound value of the distribution.

Definition at line 4130 of file random.h.

```
5.889.3.4 template<typename _IntType > template<typename _UniformRandomNumberGenerator >
negative_binomial_distribution< _IntType >::result_type std::negative_binomial_distribution< _IntType
>::operator() ( _UniformRandomNumberGenerator & __urng )
```

Generating functions.

Definition at line 1126 of file bits/random.tcc.

References std::exp(), std::basic_ios< _CharT, _Traits >::fill(), std::ios_base::flags(), std::left(), std::log(), std::min(), std::poisson_distribution< _IntType >::operator>(), std::__detail::operator>>(), std::negative_binomial_distribution< _IntType >::param(), std::ios_base::precision(), std::scientific(), std::skipws(), std::sqrt(), and std::basic_ios< _CharT, _Traits >::widen().

Referenced by std::operator>>().

```
5.889.3.5 template<typename _IntType = int> double std::negative_binomial_distribution< _IntType >::p ( ) const
[inline]
```

Return the p parameter of the distribution.

Definition at line 4108 of file random.h.

```
5.889.3.6 template<typename _IntType = int> param_type std::negative_binomial_distribution< _IntType >::param ( )
const [inline]
```

Returns the parameter set of the distribution.

Definition at line 4115 of file random.h.

Referenced by std::negative_binomial_distribution< _IntType >::operator()().

```
5.889.3.7 template<typename _IntType = int> void std::negative_binomial_distribution< _IntType >::param ( const
param_type & __param ) [inline]
```

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

Definition at line 4123 of file random.h.

5.889.3.8 `template<typename _IntType = int> void std::negative_binomial_distribution< _IntType >::reset ()`
`[inline]`

Resets the distribution state.

Definition at line 4094 of file random.h.

5.889.4 Friends And Related Function Documentation

5.889.4.1 `template<typename _IntType = int> template<typename _IntType1 , typename _CharT , typename _Traits >`
`std::basic_ostream< _CharT, _Traits>& operator<< (std::basic_ostream< _CharT, _Traits > & __os, const`
`std::negative_binomial_distribution< _IntType1 > & __x) [friend]`

Inserts a negative_binomial_distribution random number distribution __x into the output stream __os.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A negative_binomial_distribution random number distribution.

Returns

The output stream with the state of __x inserted or in an error state.

5.889.4.2 `template<typename _IntType = int> bool operator== (const negative_binomial_distribution< _IntType > & __d1,`
`const negative_binomial_distribution< _IntType > & __d2) [friend]`

Return true if two negative binomial distributions have the same parameters and the sequences that would be generated are equal.

Definition at line 4186 of file random.h.

5.889.4.3 `template<typename _IntType = int> template<typename _IntType1 , typename _CharT , typename _Traits >`
`std::basic_istream< _CharT, _Traits>& operator>> (std::basic_istream< _CharT, _Traits > & __is,`
`std::negative_binomial_distribution< _IntType1 > & __x) [friend]`

Extracts a negative_binomial_distribution random number distribution __x from the input stream __is.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A negative_binomial_distribution random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.890 `std::negative_binomial_distribution<_IntType>::param_type` Struct Reference

Public Types

- typedef [negative_binomial_distribution<_IntType>](#) **distribution_type**

Public Member Functions

- **param_type** (`_IntType __k=1`, `double __p=0.5`)
- `_IntType k` () const
- `double p` () const

Friends

- `bool operator==` (const [param_type](#) &__p1, const [param_type](#) &__p2)

5.890.1 Detailed Description

```
template<typename _IntType = int>
struct std::negative_binomial_distribution<_IntType>::param_type
```

Parameter type.

Definition at line 4052 of file `random.h`.

The documentation for this struct was generated from the following file:

- [random.h](#)

5.891 `std::nested_exception` Class Reference

Inherited by `std::_Nested_exception<_Except>`.

Public Member Functions

- **nested_exception** (const [nested_exception](#) &) noexcept=default
- exception_ptr **nested_ptr** () const noexcept
- [nested_exception](#) & **operator=** (const [nested_exception](#) &) noexcept=default
- void **rethrow_nested** () const

5.891.1 Detailed Description

Exception class with exception_ptr data member.

Definition at line 56 of file nested_exception.h.

The documentation for this class was generated from the following file:

- [nested_exception.h](#)

5.892 std::normal_distribution<_RealType> Class Template Reference

Classes

- struct [param_type](#)

Public Types

- typedef _RealType [result_type](#)

Public Member Functions

- [normal_distribution](#) ([result_type](#) __mean=[result_type](#)(0), [result_type](#) __stddev=[result_type](#)(1))
- **normal_distribution** (const [param_type](#) &__p)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- template<typename _UniformRandomNumberGenerator >
void **generate** ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [result_type](#) **max** () const
- _RealType **mean** () const
- [result_type](#) **min** () const
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [param_type](#) **param** () const
- void **param** (const [param_type](#) &__param)
- void **reset** ()
- _RealType **stddev** () const

Friends

- template<typename _RealType1, typename _CharT, typename _Traits>
std::basic_ostream<_CharT, _Traits> & operator<< (std::basic_ostream<_CharT, _Traits> &__os, const std::normal_distribution<_RealType1> &__x)
- template<typename _RealType1>
bool operator==(const std::normal_distribution<_RealType1> &__d1, const std::normal_distribution<_RealType1> &__d2)
- template<typename _RealType1, typename _CharT, typename _Traits>
std::basic_istream<_CharT, _Traits> & operator>> (std::basic_istream<_CharT, _Traits> &__is, std::normal_distribution<_RealType1> &__x)

5.892.1 Detailed Description

```
template<typename _RealType = double>
class std::normal_distribution<_RealType>
```

A normal continuous distribution for random numbers.

The formula for the normal probability density function is

$$p(x|\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x-\mu}{2\sigma^2}}$$

Definition at line 1920 of file random.h.

5.892.2 Member Typedef Documentation

5.892.2.1 template<typename _RealType = double> typedef _RealType std::normal_distribution<_RealType>::result_type

The type of the range of the distribution.

Definition at line 1923 of file random.h.

5.892.3 Constructor & Destructor Documentation

5.892.3.1 template<typename _RealType = double> std::normal_distribution<_RealType>::normal_distribution (result_type __mean = result_type(0), result_type __stddev = result_type(1)) [inline], [explicit]

Constructs a normal distribution with parameters *mean* and standard deviation.

Definition at line 1965 of file random.h.

5.892.4 Member Function Documentation

5.892.4.1 `template<typename _RealType = double> result_type std::normal_distribution<_RealType>::max () const`
`[inline]`

Returns the least upper bound value of the distribution.

Definition at line 2022 of file random.h.

5.892.4.2 `template<typename _RealType = double> _RealType std::normal_distribution<_RealType>::mean () const`
`[inline]`

Returns the mean of the distribution.

Definition at line 1986 of file random.h.

5.892.4.3 `template<typename _RealType = double> result_type std::normal_distribution<_RealType>::min () const`
`[inline]`

Returns the greatest lower bound value of the distribution.

Definition at line 2015 of file random.h.

5.892.4.4 `template<typename _RealType = double> template<typename _UniformRandomNumberGenerator> result_type`
`std::normal_distribution<_RealType>::operator() (_UniformRandomNumberGenerator & __urng) [inline]`

Generating functions.

Definition at line 2030 of file random.h.

Referenced by `std::operator>>()`.

5.892.4.5 `template<typename _RealType> template<typename _UniformRandomNumberGenerator>`
`normal_distribution<_RealType>::result_type std::normal_distribution<_RealType>::operator() (`
`_UniformRandomNumberGenerator & __urng, const param_type & __param)`

Polar method due to Marsaglia.

Devroye, L. Non-Uniform Random Variates Generation. Springer-Verlag, New York, 1986, Ch. V, Sect. 4.4.

Definition at line 1783 of file bits/random.tcc.

References `std::dec()`, `std::exp()`, `std::basic_ios<_CharT, _Traits>::fill()`, `std::ios_base::flags()`, `std::left()`, `std::log()`,
`std::cauchy_distribution<_RealType>::operator()()`, `std::__detail::operator>>()`, `std::normal_distribution<_RealType>`
`>::param()`, `std::lognormal_distribution<_RealType>::param()`, `std::chi_squared_distribution<_RealType>::param()`,
`std::ios_base::precision()`, `std::scientific()`, `std::skipws()`, `std::sqrt()`, `std::tan()`, and `std::basic_ios<_CharT, _Traits>::widen()`.

5.892.4.6 `template<typename _RealType = double> param_type std::normal_distribution<_RealType>::param () const`
`[inline]`

Returns the parameter set of the distribution.

Definition at line 2000 of file random.h.

Referenced by `std::normal_distribution<_RealType>::operator()()`.

5.892.4.7 `template<typename _RealType = double> void std::normal_distribution<_RealType>::param (const`
`param_type & __param) [inline]`

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

Definition at line 2008 of file random.h.

5.892.4.8 `template<typename _RealType = double> void std::normal_distribution<_RealType>::reset() [inline]`

Resets the distribution state.

Definition at line 1979 of file random.h.

5.892.4.9 `template<typename _RealType = double> _RealType std::normal_distribution<_RealType>::stddev() const [inline]`

Returns the standard deviation of the distribution.

Definition at line 1993 of file random.h.

5.892.5 Friends And Related Function Documentation

5.892.5.1 `template<typename _RealType = double> template<typename _RealType1, typename _CharT, typename _Traits> std::basic_ostream<_CharT, _Traits>& operator<<(std::basic_ostream<_CharT, _Traits> & __os, const std::normal_distribution<_RealType1> & __x) [friend]`

Inserts a normal_distribution random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A normal_distribution random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

5.892.5.2 `template<typename _RealType = double> template<typename _RealType1> bool operator==(const std::normal_distribution<_RealType1> & __d1, const std::normal_distribution<_RealType1> & __d2) [friend]`

Return true if two normal distributions have the same parameters and the sequences that would be generated are equal.

5.892.5.3 `template<typename _RealType = double> template<typename _RealType1, typename _CharT, typename _Traits> std::basic_istream<_CharT, _Traits>& operator>>(std::basic_istream<_CharT, _Traits> & __is, std::normal_distribution<_RealType1> & __x) [friend]`

Extracts a normal_distribution random number distribution `__x` from the input stream `__is`.

Parameters

<code>_↔ _is</code>	An input stream.
<code>_↔ _x</code>	A normal_distribution random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.893 `std::normal_distribution<_RealType>::param_type` Struct Reference

Public Types

- typedef `normal_distribution<_RealType>` **distribution_type**

Public Member Functions

- **param_type** (`_RealType __mean=_RealType(0), _RealType __stddev=_RealType(1)`)
- `_RealType mean` () const
- `_RealType stddev` () const

Friends

- bool **operator==** (const `param_type` &__p1, const `param_type` &__p2)

5.893.1 Detailed Description

```
template<typename _RealType = double>
struct std::normal_distribution<_RealType>::param_type
```

Parameter type.

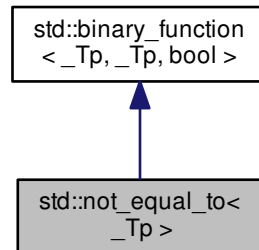
Definition at line 1929 of file random.h.

The documentation for this struct was generated from the following file:

- [random.h](#)

5.894 `std::not_equal_to<_Tp>` Struct Template Reference

Inheritance diagram for `std::not_equal_to<_Tp>`:



Public Types

- typedef `_Tp` `first_argument_type`
- typedef `bool` `result_type`
- typedef `_Tp` `second_argument_type`

Public Member Functions

- `_GLIBCXX14_CONSTEXPR bool operator() (const _Tp &__x, const _Tp &__y) const`

5.894.1 Detailed Description

```
template<typename _Tp>
struct std::not_equal_to<_Tp>
```

One of the [comparison functors](#).

Definition at line 334 of file `stl_function.h`.

5.894.2 Member Typedef Documentation

5.894.2.1 typedef `_Tp` `std::binary_function<_Tp, _Tp, bool>::first_argument_type` [\[inherited\]](#)

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.894.2.2 `typedef bool std::binary_function<_Tp, _Tp, bool>::result_type` [inherited]

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.894.2.3 `typedef _Tp std::binary_function<_Tp, _Tp, bool>::second_argument_type` [inherited]

`second_argument_type` is the type of the second argument

Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.895 `std::not_equal_to< void >` Struct Template Reference

Public Types

- `typedef __is_transparent is_transparent`

Public Member Functions

- `template<typename _Tp, typename _Up >
_GLIBCXX14_CONSTEXPR auto operator() (_Tp &&__t, _Up &&__u) const noexcept(noexcept(std::forward<
_Tp >(__t)!=std::forward<_Up >(__u))) -> decltype(std::forward<_Tp >(__t)!=std::forward<_Up >(__u))`

5.895.1 Detailed Description

```
template<>
struct std::not_equal_to< void >
```

One of the [comparison functors](#).

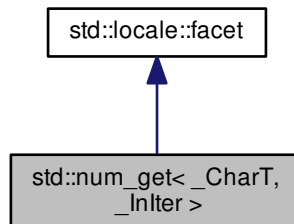
Definition at line 427 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.896 std::num_get<_CharT, _InIter> Class Template Reference

Inheritance diagram for std::num_get<_CharT, _InIter>:



Public Types

- typedef _CharT [char_type](#)
- typedef _InIter [iter_type](#)

Public Member Functions

- [num_get](#) (size_t __refs=0)
- template<typename _ValueT>
_GLIBCXX_DEFAULT_ABI_TAG _InIter **M_extract_int** (_InIter __beg, _InIter __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, _ValueT &__v) const
- [iter_type](#) get ([iter_type](#) __in, [iter_type](#) __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, bool &__v) const
- [iter_type](#) get ([iter_type](#) __in, [iter_type](#) __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, void *&__v) const
- [iter_type](#) get ([iter_type](#) __in, [iter_type](#) __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, long &__v) const
- [iter_type](#) get ([iter_type](#) __in, [iter_type](#) __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, unsigned short &__v) const
- [iter_type](#) get ([iter_type](#) __in, [iter_type](#) __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, unsigned int &__v) const
- [iter_type](#) get ([iter_type](#) __in, [iter_type](#) __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, unsigned long &__v) const
- [iter_type](#) get ([iter_type](#) __in, [iter_type](#) __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, long long &__v) const
- [iter_type](#) get ([iter_type](#) __in, [iter_type](#) __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, unsigned long long &__v) const
- [iter_type](#) get ([iter_type](#) __in, [iter_type](#) __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, float &__v) const
- [iter_type](#) get ([iter_type](#) __in, [iter_type](#) __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, double &__v) const
- [iter_type](#) get ([iter_type](#) __in, [iter_type](#) __end, [ios_base](#) &__io, [ios_base::iostate](#) &__err, long double &__v) const

Static Public Attributes

- static [locale::id](#) id

Protected Member Functions

- virtual [~num_get](#) ()
- [_GLIBCXX_DEFAULT_ABI_TAG iter_type _M_extract_float](#) ([iter_type](#), [iter_type](#), [ios_base &](#), [ios_base::iostate &](#), [string](#) &) const
- template<typename [_ValueT](#) >
[_GLIBCXX_DEFAULT_ABI_TAG iter_type _M_extract_int](#) ([iter_type](#), [iter_type](#), [ios_base &](#), [ios_base::iostate &](#), [_ValueT](#) &) const
- template<typename [_CharT2](#) >
[__gnu_cxx::__enable_if< __is_char< \[_CharT2\]\(#\) >::__value, int >::__type](#) [_M_find](#) (const [_CharT2](#) *, [size_t](#) __len, [_CharT2](#) __c) const
- template<typename [_CharT2](#) >
[__gnu_cxx::__enable_if<! __is_char< \[_CharT2\]\(#\) >::__value, int >::__type](#) [_M_find](#) (const [_CharT2](#) * __zero, [size_t](#) __len, [_CharT2](#) __c) const
- virtual [iter_type do_get](#) ([iter_type](#), [iter_type](#), [ios_base &](#), [ios_base::iostate &](#), bool &) const
- virtual [iter_type do_get](#) ([iter_type](#) __beg, [iter_type](#) __end, [ios_base &](#) __io, [ios_base::iostate &](#) __err, long & __v) const
- virtual [iter_type do_get](#) ([iter_type](#) __beg, [iter_type](#) __end, [ios_base &](#) __io, [ios_base::iostate &](#) __err, unsigned short & __v) const
- virtual [iter_type do_get](#) ([iter_type](#) __beg, [iter_type](#) __end, [ios_base &](#) __io, [ios_base::iostate &](#) __err, unsigned int & __v) const
- virtual [iter_type do_get](#) ([iter_type](#) __beg, [iter_type](#) __end, [ios_base &](#) __io, [ios_base::iostate &](#) __err, unsigned long & __v) const
- virtual [iter_type do_get](#) ([iter_type](#) __beg, [iter_type](#) __end, [ios_base &](#) __io, [ios_base::iostate &](#) __err, long long & __v) const
- virtual [iter_type do_get](#) ([iter_type](#) __beg, [iter_type](#) __end, [ios_base &](#) __io, [ios_base::iostate &](#) __err, unsigned long long & __v) const
- virtual [iter_type do_get](#) ([iter_type](#), [iter_type](#), [ios_base &](#), [ios_base::iostate &](#), float &) const
- virtual [iter_type do_get](#) ([iter_type](#), [iter_type](#), [ios_base &](#), [ios_base::iostate &](#), double &) const
- virtual [iter_type do_get](#) ([iter_type](#), [iter_type](#), [ios_base &](#), [ios_base::iostate &](#), long double &) const
- virtual [iter_type do_get](#) ([iter_type](#), [iter_type](#), [ios_base &](#), [ios_base::iostate &](#), void *&) const

Static Protected Member Functions

- static [__c_locale _S_clone_c_locale](#) ([__c_locale &](#) __cloc) throw ()
- static void [_S_create_c_locale](#) ([__c_locale &](#) __cloc, const char * __s, [__c_locale](#) __old=0)
- static void [_S_destroy_c_locale](#) ([__c_locale &](#) __cloc)
- static [__c_locale _S_get_c_locale](#) ()
- static const char * [_S_get_c_name](#) () throw ()
- static [__c_locale _S_lc_ctype_c_locale](#) ([__c_locale](#) __cloc, const char * __s)

5.896.1 Detailed Description

```
template<typename _CharT, typename _InIter>  
class std::num_get<_CharT, _InIter>
```

Primary class template num_get.

This facet encapsulates the code to parse and return a number from a string. It is used by the istream numeric extraction operators.

The num_get template uses protected virtual functions to provide the actual results. The public accessors forward the call to the virtual functions. These virtual functions are hooks for developers to implement the behavior they require from the num_get facet.

Definition at line 1948 of file locale_facets.h.

5.896.2 Member Typedef Documentation

5.896.2.1 `template<typename _CharT, typename _InIter> typedef _CharT std::num_get<_CharT, _InIter>::char_type`

Public typedefs.

Definition at line 1954 of file locale_facets.h.

5.896.2.2 `template<typename _CharT, typename _InIter> typedef _InIter std::num_get<_CharT, _InIter>::iter_type`

Public typedefs.

Definition at line 1955 of file locale_facets.h.

5.896.3 Constructor & Destructor Documentation

5.896.3.1 `template<typename _CharT, typename _InIter> std::num_get<_CharT, _InIter>::num_get (size_t __refs = 0)
[inline], [explicit]`

Constructor performs initialization.

This is the constructor provided by the standard.

Parameters

<code>__refs</code>	Passed to the base facet class.
---------------------	---------------------------------

Definition at line 1969 of file locale_facets.h.

5.896.3.2 `template<typename _CharT, typename _Inlter > virtual std::num_get< _CharT, _Inlter >::~~num_get ()`
`[inline], [protected], [virtual]`

Destructor.

Definition at line 2141 of file locale_facets.h.

5.896.4 Member Function Documentation

5.896.4.1 `template<typename _CharT, typename _Inlter > _Inlter std::num_get< _CharT, _Inlter >::do_get (iter_type`
`__beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, bool & __v) const` `[protected],`
`[virtual]`

Numeric parsing.

Parses the input stream into the variable *v*. This function is a hook for derived classes to change the value returned.

See also

`get()` for more details.

Parameters

<code>__beg</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 595 of file locale_facets.tcc.

References `std::ios_base::_M_getloc()`, `std::ios_base::boolalpha`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::flags()`, and `std::ios_base::goodbit`.

Referenced by `std::num_get< _CharT, _Inlter >::do_get()`.

5.896.4.2 `template<typename _CharT, typename _Inlter > virtual iter_type std::num_get< _CharT, _Inlter >::do_get (`
`iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, long & __v) const` `[inline],`
`[protected], [virtual]`

Numeric parsing.

Parses the input stream into the variable *v*. This function is a hook for derived classes to change the value returned.

See also

`get()` for more details.

Parameters

<code>__beg</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 2211 of file locale_facets.h.

```
5.896.4.3 template<typename _CharT, typename _InIter > virtual iter_type std::num_get<_CharT, _InIter >::do_get (
    iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, unsigned short & __v ) const
    [inline], [protected], [virtual]
```

Numeric parsing.

Parses the input stream into the variable `v`. This function is a hook for derived classes to change the value returned.

See also

`get()` for more details.

Parameters

<code>__beg</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 2216 of file locale_facets.h.

```
5.896.4.4 template<typename _CharT, typename _InIter > virtual iter_type std::num_get<_CharT, _InIter >::do_get (
    iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, unsigned int & __v ) const
    [inline], [protected], [virtual]
```

Numeric parsing.

Parses the input stream into the variable `v`. This function is a hook for derived classes to change the value returned.

See also

`get()` for more details.

Parameters

<code>__beg</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 2221 of file `locale_facets.h`.

```
5.896.4.5 template<typename _CharT, typename _InIter > virtual iter_type std::num_get< _CharT, _InIter >::do_get (
    iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, unsigned long & __v ) const
    [inline], [protected], [virtual]
```

Numeric parsing.

Parses the input stream into the variable `v`. This function is a hook for derived classes to change the value returned.

See also

`get()` for more details.

Parameters

<code>__beg</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 2226 of file `locale_facets.h`.

5.896.4.6 `template<typename _CharT, typename _InIter > virtual iter_type std::num_get<_CharT, _InIter >::do_get (iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, long long & __v) const [inline], [protected], [virtual]`

Numeric parsing.

Parses the input stream into the variable *v*. This function is a hook for derived classes to change the value returned.

See also

`get()` for more details.

Parameters

<code>__beg</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 2232 of file `locale_facets.h`.

5.896.4.7 `template<typename _CharT, typename _InIter > virtual iter_type std::num_get<_CharT, _InIter >::do_get (iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, unsigned long long & __v) const [inline], [protected], [virtual]`

Numeric parsing.

Parses the input stream into the variable *v*. This function is a hook for derived classes to change the value returned.

See also

`get()` for more details.

Parameters

<code>__beg</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 2237 of file locale_facets.h.

```
5.896.4.8 template<typename _CharT, typename _InIter > _InIter std::num_get< _CharT, _InIter >::do_get ( iter_type
    __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, float & __v ) const    [protected],
    [virtual]
```

Numeric parsing.

Parses the input stream into the variable `v`. This function is a hook for derived classes to change the value returned.

See also

`get()` for more details.

Parameters

<code>__beg</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 691 of file locale_facets.tcc.

References `std::basic_string< _CharT, _Traits, _Alloc >::c_str()`, `std::num_get< _CharT, _InIter >::do_get()`, `std::ios_base::eofbit`, and `std::basic_string< _CharT, _Traits, _Alloc >::reserve()`.

```
5.896.4.9 template<typename _CharT, typename _InIter > _InIter std::num_get< _CharT, _InIter >::do_get ( iter_type __beg,
    iter_type __end, ios_base & __io, ios_base::iostate & __err, double & __v ) const    [protected],
    [virtual]
```

Numeric parsing.

Parses the input stream into the variable `v`. This function is a hook for derived classes to change the value returned.

See also

`get()` for more details.

Parameters

<code>__beg</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 706 of file locale_facets.tcc.

References `std::basic_string< _CharT, _Traits, _Alloc >::c_str()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::ios_base::eofbit`, and `std::basic_string< _CharT, _Traits, _Alloc >::reserve()`.

5.896.4.10 `template<typename _CharT, typename _Inlter > _Inlter std::num_get< _CharT, _Inlter >::do_get(iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, long double & __v) const [protected], [virtual]`

Numeric parsing.

Parses the input stream into the variable `v`. This function is a hook for derived classes to change the value returned.

See also

`get()` for more details.

Parameters

<code>__beg</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 738 of file locale_facets.tcc.

References `std::basic_string< _CharT, _Traits, _Alloc >::c_str()`, `std::num_get< _CharT, _Inlter >::do_get()`, `std::ios_base::eofbit`, and `std::basic_string< _CharT, _Traits, _Alloc >::reserve()`.

5.896.4.11 `template<typename _CharT, typename _InIter > _InIter std::num_get< _CharT, _InIter >::do_get(iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, void *& __v) const` [protected], [virtual]

Numeric parsing.

Parses the input stream into the variable *v*. This function is a hook for derived classes to change the value returned.

See also

`get()` for more details.

Parameters

<code>__beg</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 753 of file `locale_facets.tcc`.

References `std::ios_base::_M_getloc()`, `std::ios_base::basefield`, `std::num_put< _CharT, _OutIter >::do_put()`, `std::ios_base::fixed`, `std::ios_base::flags()`, `std::ios_base::floatfield`, `std::ios_base::hex`, `std::ios_base::oct`, `std::ios_base::precision()`, `std::ios_base::showbase`, `std::ios_base::showpos`, `std::ios_base::uppercase`, `std::ctype_abstract_base< _CharT >::widen()`, and `std::ios_base::width()`.

5.896.4.12 `template<typename _CharT, typename _InIter > iter_type std::num_get< _CharT, _InIter >::get(iter_type __in, iter_type __end, ios_base & __io, ios_base::iostate & __err, bool & __v) const` [inline]

Numeric parsing.

Parses the input stream into the bool *v*. It does so by calling `num_get::do_get()`.

If `ios_base::boolalpha` is set, attempts to read `ctype<CharT>::truenamename()` or `ctype<CharT>::falsename()`. Sets *v* to true or false if successful. Sets *err* to `ios_base::failbit` if reading the string fails. Sets *err* to `ios_base::eofbit` if the stream is emptied.

If `ios_base::boolalpha` is not set, proceeds as with reading a long, except if the value is 1, sets *v* to true, if the value is 0, sets *v* to false, and otherwise set *err* to `ios_base::failbit`.

Parameters

<code>__in</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 1995 of file locale_facets.h.

Referenced by std::basic_istream< _CharT, _Traits >::operator>>(), and std::basic_istream< _CharT, _Traits >::sentry::sentry().

5.896.4.13 `template<typename _CharT, typename _InIter> iter_type std::num_get< _CharT, _InIter >::get (iter_type __in, iter_type __end, ios_base & __io, ios_base::iostate & __err, long & __v) const [inline]`

Numeric parsing.

Parses the input stream into the integral variable *v*. It does so by calling num_get::do_get().

Parsing is affected by the flag settings in *io*.

The basic parse is affected by the value of io.flags() & ios_base::basefield. If equal to ios_base::oct, parses like the scanf o specifier. Else if equal to ios_base::hex, parses like X specifier. Else if basefield equal to 0, parses like the i specifier. Otherwise, parses like d for signed and u for unsigned types. The matching type length modifier is also used.

Digit grouping is interpreted according to numpunct::grouping() and numpunct::thousands_sep(). If the pattern of digit groups isn't consistent, sets err to ios_base::failbit.

If parsing the string yields a valid value for *v*, *v* is set. Otherwise, sets err to ios_base::failbit and leaves *v* unaltered. Sets err to ios_base::eofbit if the stream is emptied.

Parameters

<code>__in</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 2032 of file locale_facets.h.

5.896.4.14 `template<typename _CharT, typename _InIter> iter_type std::num_get< _CharT, _InIter >::get (iter_type __in, iter_type __end, ios_base & __io, ios_base::iostate & __err, unsigned short & __v) const [inline]`

Numeric parsing.

Parses the input stream into the integral variable *v*. It does so by calling num_get::do_get().

Parsing is affected by the flag settings in *io*.

The basic parse is affected by the value of `io.flags()` & `ios_base::basefield`. If equal to `ios_base::oct`, parses like the `scanf` `o` specifier. Else if equal to `ios_base::hex`, parses like `X` specifier. Else if `basefield` equal to 0, parses like the `i` specifier. Otherwise, parses like `d` for signed and `u` for unsigned types. The matching type length modifier is also used.

Digit grouping is interpreted according to `numpunct::grouping()` and `numpunct::thousands_sep()`. If the pattern of digit groups isn't consistent, sets `err` to `ios_base::failbit`.

If parsing the string yields a valid value for `v`, `v` is set. Otherwise, sets `err` to `ios_base::failbit` and leaves `v` unaltered. Sets `err` to `ios_base::eofbit` if the stream is emptied.

Parameters

<code>__in</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 2037 of file `locale_facets.h`.

```
5.896.4.15  template<typename _CharT, typename _InIter > iter_type std::num_get< _CharT, _InIter >::get( iter_type __in,
iter_type __end, ios_base & __io, ios_base::iostate & __err, unsigned int & __v ) const  [inline]
```

Numeric parsing.

Parses the input stream into the integral variable `v`. It does so by calling `num_get::do_get()`.

Parsing is affected by the flag settings in `io`.

The basic parse is affected by the value of `io.flags()` & `ios_base::basefield`. If equal to `ios_base::oct`, parses like the `scanf` `o` specifier. Else if equal to `ios_base::hex`, parses like `X` specifier. Else if `basefield` equal to 0, parses like the `i` specifier. Otherwise, parses like `d` for signed and `u` for unsigned types. The matching type length modifier is also used.

Digit grouping is interpreted according to `numpunct::grouping()` and `numpunct::thousands_sep()`. If the pattern of digit groups isn't consistent, sets `err` to `ios_base::failbit`.

If parsing the string yields a valid value for `v`, `v` is set. Otherwise, sets `err` to `ios_base::failbit` and leaves `v` unaltered. Sets `err` to `ios_base::eofbit` if the stream is emptied.

Parameters

<code>__in</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 2042 of file locale_facets.h.

```
5.896.4.16 template<typename _CharT, typename _InIter > iter_type std::num_get<_CharT, _InIter >::get( iter_type __in,
iter_type __end, ios_base & __io, ios_base::iostate & __err, unsigned long & __v ) const [inline]
```

Numeric parsing.

Parses the input stream into the integral variable *v*. It does so by calling num_get::do_get().

Parsing is affected by the flag settings in *io*.

The basic parse is affected by the value of io.flags() & ios_base::basefield. If equal to ios_base::oct, parses like the scanf o specifier. Else if equal to ios_base::hex, parses like X specifier. Else if basefield equal to 0, parses like the i specifier. Otherwise, parses like d for signed and u for unsigned types. The matching type length modifier is also used.

Digit grouping is interpreted according to numpunct::grouping() and numpunct::thousands_sep(). If the pattern of digit groups isn't consistent, sets err to ios_base::failbit.

If parsing the string yields a valid value for *v*, *v* is set. Otherwise, sets err to ios_base::failbit and leaves *v* unaltered. Sets err to ios_base::eofbit if the stream is emptied.

Parameters

<code>__in</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 2047 of file locale_facets.h.

```
5.896.4.17 template<typename _CharT, typename _InIter > iter_type std::num_get<_CharT, _InIter >::get( iter_type __in,
iter_type __end, ios_base & __io, ios_base::iostate & __err, long long & __v ) const [inline]
```

Numeric parsing.

Parses the input stream into the integral variable *v*. It does so by calling num_get::do_get().

Parsing is affected by the flag settings in *io*.

The basic parse is affected by the value of io.flags() & ios_base::basefield. If equal to ios_base::oct, parses like the scanf o specifier. Else if equal to ios_base::hex, parses like X specifier. Else if basefield equal to 0, parses like the i specifier. Otherwise, parses like d for signed and u for unsigned types. The matching type length modifier is also used.

Digit grouping is interpreted according to `numpunct::grouping()` and `numpunct::thousands_sep()`. If the pattern of digit groups isn't consistent, sets `err` to `ios_base::failbit`.

If parsing the string yields a valid value for `v`, `v` is set. Otherwise, sets `err` to `ios_base::failbit` and leaves `v` unaltered. Sets `err` to `ios_base::eofbit` if the stream is emptied.

Parameters

<code>__in</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 2053 of file `locale_facets.h`.

```
5.896.4.18  template<typename _CharT, typename _InIter > iter_type std::num_get< _CharT, _InIter >::get ( iter_type __in,
            iter_type __end, ios_base & __io, ios_base::iostate & __err, unsigned long long & __v ) const  [inline]
```

Numeric parsing.

Parses the input stream into the integral variable `v`. It does so by calling `num_get::do_get()`.

Parsing is affected by the flag settings in `io`.

The basic parse is affected by the value of `io.flags()` & `ios_base::basefield`. If equal to `ios_base::oct`, parses like the `scanf` `o` specifier. Else if equal to `ios_base::hex`, parses like `X` specifier. Else if `basefield` equal to 0, parses like the `i` specifier. Otherwise, parses like `d` for signed and `u` for unsigned types. The matching type length modifier is also used.

Digit grouping is interpreted according to `numpunct::grouping()` and `numpunct::thousands_sep()`. If the pattern of digit groups isn't consistent, sets `err` to `ios_base::failbit`.

If parsing the string yields a valid value for `v`, `v` is set. Otherwise, sets `err` to `ios_base::failbit` and leaves `v` unaltered. Sets `err` to `ios_base::eofbit` if the stream is emptied.

Parameters

<code>__in</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 2058 of file locale_facets.h.

```
5.896.4.19  template<typename _CharT, typename _InIter> iter_type std::num_get<_CharT, _InIter>::get( iter_type __in,
iter_type __end, ios_base & __io, ios_base::iostate & __err, float & __v ) const  [inline]
```

Numeric parsing.

Parses the input stream into the integral variable *v*. It does so by calling num_get::do_get().

The input characters are parsed like the scanf g specifier. The matching type length modifier is also used.

The decimal point character used is numpunct::decimal_point(). Digit grouping is interpreted according to numpunct::grouping() and numpunct::thousands_sep(). If the pattern of digit groups isn't consistent, sets err to ios_base::failbit.

If parsing the string yields a valid value for *v*, *v* is set. Otherwise, sets err to ios_base::failbit and leaves *v* unaltered. Sets err to ios_base::eofbit if the stream is emptied.

Parameters

<code>__in</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 2092 of file locale_facets.h.

```
5.896.4.20  template<typename _CharT, typename _InIter> iter_type std::num_get<_CharT, _InIter>::get( iter_type __in,
iter_type __end, ios_base & __io, ios_base::iostate & __err, double & __v ) const  [inline]
```

Numeric parsing.

Parses the input stream into the integral variable *v*. It does so by calling num_get::do_get().

The input characters are parsed like the scanf g specifier. The matching type length modifier is also used.

The decimal point character used is numpunct::decimal_point(). Digit grouping is interpreted according to numpunct::grouping() and numpunct::thousands_sep(). If the pattern of digit groups isn't consistent, sets err to ios_base::failbit.

If parsing the string yields a valid value for *v*, *v* is set. Otherwise, sets err to ios_base::failbit and leaves *v* unaltered. Sets err to ios_base::eofbit if the stream is emptied.

Parameters

<code>__in</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 2097 of file `locale_facets.h`.

```
5.896.4.21  template<typename _CharT, typename _InIter > iter_type std::num_get<_CharT, _InIter>::get( iter_type __in,
iter_type __end, ios_base & __io, ios_base::iostate & __err, long double & __v ) const  [inline]
```

Numeric parsing.

Parses the input stream into the integral variable `v`. It does so by calling `num_get::do_get()`.

The input characters are parsed like the `scanf g` specifier. The matching type length modifier is also used.

The decimal point character used is `num_punct::decimal_point()`. Digit grouping is interpreted according to `num_punct::grouping()` and `num_punct::thousands_sep()`. If the pattern of digit groups isn't consistent, sets `err` to `ios_base::failbit`.

If parsing the string yields a valid value for `v`, `v` is set. Otherwise, sets `err` to `ios_base::failbit` and leaves `v` unaltered. Sets `err` to `ios_base::eofbit` if the stream is emptied.

Parameters

<code>__in</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 2102 of file `locale_facets.h`.

```
5.896.4.22  template<typename _CharT, typename _InIter > iter_type std::num_get<_CharT, _InIter>::get( iter_type __in,
iter_type __end, ios_base & __io, ios_base::iostate & __err, void *& __v ) const  [inline]
```

Numeric parsing.

Parses the input stream into the pointer variable *v*. It does so by calling `num_get::do_get()`.

The input characters are parsed like the `scanf p` specifier.

Digit grouping is interpreted according to `num_punct::grouping()` and `num_punct::thousands_sep()`. If the pattern of digit groups isn't consistent, sets `err` to `ios_base::failbit`.

Note that the digit grouping effect for pointers is a bit ambiguous in the standard and shouldn't be relied on. See DR 344.

If parsing the string yields a valid value for *v*, *v* is set. Otherwise, sets `err` to `ios_base::failbit` and leaves *v* unaltered. Sets `err` to `ios_base::eofbit` if the stream is emptied.

Parameters

<code>__in</code>	Start of input stream.
<code>__end</code>	End of input stream.
<code>__io</code>	Source of locale and flags.
<code>__err</code>	Error flags to set.
<code>__v</code>	Value to format and insert.

Returns

Iterator after reading.

Definition at line 2135 of file `locale_facets.h`.

5.896.5 Member Data Documentation

5.896.5.1 `template<typename _CharT, typename _InIter> locale::id std::num_get<_CharT, _InIter>::id` `[static]`

Numpunct facet id.

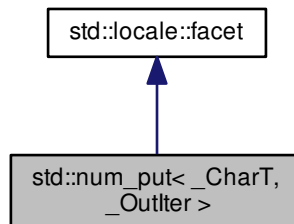
Definition at line 1959 of file `locale_facets.h`.

The documentation for this class was generated from the following files:

- [locale_facets.h](#)
- [locale_facets.tcc](#)

5.897 std::num_put<_CharT, _Outlter> Class Template Reference

Inheritance diagram for std::num_put<_CharT, _Outlter>:



Public Types

- typedef `_CharT` `char_type`
- typedef `_Outlter` `iter_type`

Public Member Functions

- `num_put` (`size_t __refs=0`)
- `template<typename _ValueT>`
`_Outlter _M_insert_float` (`_Outlter __s, ios_base &__io, _CharT __fill, char __mod, _ValueT __v`) `const`
- `template<typename _ValueT>`
`_Outlter _M_insert_int` (`_Outlter __s, ios_base &__io, _CharT __fill, _ValueT __v`) `const`
- `iter_type put` (`iter_type __s, ios_base &__io, char_type __fill, bool __v`) `const`
- `iter_type put` (`iter_type __s, ios_base &__io, char_type __fill, const void *__v`) `const`
- `iter_type put` (`iter_type __s, ios_base &__io, char_type __fill, long __v`) `const`
- `iter_type put` (`iter_type __s, ios_base &__io, char_type __fill, unsigned long __v`) `const`
- `iter_type put` (`iter_type __s, ios_base &__io, char_type __fill, long long __v`) `const`
- `iter_type put` (`iter_type __s, ios_base &__io, char_type __fill, unsigned long long __v`) `const`
- `iter_type put` (`iter_type __s, ios_base &__io, char_type __fill, double __v`) `const`
- `iter_type put` (`iter_type __s, ios_base &__io, char_type __fill, long double __v`) `const`

Static Public Attributes

- static `locale::id` `id`

Protected Member Functions

- virtual `~num_put()`
- void `_M_group_float` (const char * __grouping, size_t __grouping_size, char_type __sep, const char_type * __p, char_type * __new, char_type * __cs, int & __len) const
- void `_M_group_int` (const char * __grouping, size_t __grouping_size, char_type __sep, ios_base & __io, char_type * __new, char_type * __cs, int & __len) const
- template<typename _ValueT>
iter_type `_M_insert_float` (iter_type, ios_base & __io, char_type __fill, char __mod, _ValueT __v) const
- template<typename _ValueT>
iter_type `_M_insert_int` (iter_type, ios_base & __io, char_type __fill, _ValueT __v) const
- void `_M_pad` (char_type __fill, streamsize __w, ios_base & __io, char_type * __new, const char_type * __cs, int & __len) const
- virtual iter_type `do_put` (iter_type __s, ios_base & __io, char_type __fill, bool __v) const
- virtual iter_type `do_put` (iter_type __s, ios_base & __io, char_type __fill, long __v) const
- virtual iter_type `do_put` (iter_type __s, ios_base & __io, char_type __fill, unsigned long __v) const
- virtual iter_type `do_put` (iter_type __s, ios_base & __io, char_type __fill, long long __v) const
- virtual iter_type `do_put` (iter_type __s, ios_base & __io, char_type __fill, unsigned long long __v) const
- virtual iter_type `do_put` (iter_type, ios_base & __io, char_type, double) const
- virtual iter_type `do_put` (iter_type, ios_base & __io, char_type, long double) const
- virtual iter_type `do_put` (iter_type, ios_base & __io, char_type, const void *) const

Static Protected Member Functions

- static __c_locale `_S_clone_c_locale` (__c_locale & __cloc) throw ()
- static void `_S_create_c_locale` (__c_locale & __cloc, const char * __s, __c_locale __old=0)
- static void `_S_destroy_c_locale` (__c_locale & __cloc)
- static __c_locale `_S_get_c_locale` ()
- static const char * `_S_get_c_name` () throw ()
- static __c_locale `_S_lc_type_c_locale` (__c_locale __cloc, const char * __s)

5.897.1 Detailed Description

```
template<typename _CharT, typename _Outiter>
class std::num_put<_CharT, _Outiter>
```

Primary class template num_put.

This facet encapsulates the code to convert a number to a string. It is used by the ostream numeric insertion operators.

The num_put template uses protected virtual functions to provide the actual results. The public accessors forward the call to the virtual functions. These virtual functions are hooks for developers to implement the behavior they require from the num_put facet.

Definition at line 2289 of file locale_facets.h.

5.897.2 Member Typedef Documentation

5.897.2.1 `template<typename _CharT, typename _Outiter> typedef _CharT std::num_put< _CharT, _Outiter >::char_type`

Public typedefs.

Definition at line 2295 of file locale_facets.h.

5.897.2.2 `template<typename _CharT, typename _Outiter> typedef _Outiter std::num_put< _CharT, _Outiter >::iter_type`

Public typedefs.

Definition at line 2296 of file locale_facets.h.

5.897.3 Constructor & Destructor Documentation

5.897.3.1 `template<typename _CharT, typename _Outiter> std::num_put< _CharT, _Outiter >::num_put (size_t __refs = 0) [inline],[explicit]`

Constructor performs initialization.

This is the constructor provided by the standard.

Parameters

<code>__refs</code>	Passed to the base facet class.
---------------------	---------------------------------

Definition at line 2310 of file locale_facets.h.

5.897.3.2 `template<typename _CharT, typename _Outiter> virtual std::num_put< _CharT, _Outiter >::~~num_put () [inline],[protected],[virtual]`

Destructor.

Definition at line 2489 of file locale_facets.h.

5.897.4 Member Function Documentation

5.897.4.1 `template<typename _CharT, typename _Outiter> _Outiter std::num_put< _CharT, _Outiter >::do_put (iter_type __s, ios_base & __io, char_type __fill, bool __v) const [protected],[virtual]`

Numeric formatting.

These functions do the work of formatting numeric values and inserting them into a stream. This function is a hook for derived classes to change the value returned.

Parameters

\leftarrow _s	Stream to write to.
\leftarrow _io	Source of locale and flags.
\leftarrow _fill	Char_type to use for filling.
\leftarrow _v	Value to format and insert.

Returns

Iterator after writing.

Definition at line 1106 of file locale_facets.tcc.

References std::ios_base::_M_getloc(), std::ios_base::adjustfield, std::ios_base::boolalpha, std::ios_base::flags(), std::ios_base::left, and std::ios_base::width().

Referenced by std::num_get<_CharT, _InIter >::do_get(), and std::num_put<_CharT, _OutIter >::do_put().

5.897.4.2 `template<typename _CharT, typename _OutIter > virtual iter_type std::num_put<_CharT, _OutIter >::do_put
(iter_type __s, ios_base & __io, char_type __fill, long __v) const [inline], [protected],
[virtual]`

Numeric formatting.

These functions do the work of formatting numeric values and inserting them into a stream. This function is a hook for derived classes to change the value returned.

Parameters

\leftarrow _s	Stream to write to.
\leftarrow _io	Source of locale and flags.
\leftarrow _fill	Char_type to use for filling.
\leftarrow _v	Value to format and insert.

Returns

Iterator after writing.

Definition at line 2509 of file locale_facets.h.

5.897.4.3 `template<typename _CharT, typename _Outiter > virtual iter_type std::num_put<_CharT, _Outiter >::do_put (iter_type __s, ios_base & __io, char_type __fill, unsigned long __v) const [inline], [protected], [virtual]`

Numeric formatting.

These functions do the work of formatting numeric values and inserting them into a stream. This function is a hook for derived classes to change the value returned.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

Definition at line 2513 of file locale_facets.h.

5.897.4.4 `template<typename _CharT, typename _Outiter > virtual iter_type std::num_put<_CharT, _Outiter >::do_put (iter_type __s, ios_base & __io, char_type __fill, long long __v) const [inline], [protected], [virtual]`

Numeric formatting.

These functions do the work of formatting numeric values and inserting them into a stream. This function is a hook for derived classes to change the value returned.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

Definition at line 2519 of file locale_facets.h.

```
5.897.4.5 template<typename _CharT, typename _Outlter > virtual iter_type std::num_put<_CharT, _Outlter >::do_put (
    iter_type __s, ios_base & __io, char_type __fill, unsigned long long __v ) const    [inline], [protected],
    [virtual]
```

Numeric formatting.

These functions do the work of formatting numeric values and inserting them into a stream. This function is a hook for derived classes to change the value returned.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

Definition at line 2524 of file locale_facets.h.

```
5.897.4.6 template<typename _CharT, typename _Outlter > _Outlter std::num_put<_CharT, _Outlter >::do_put ( iter_type
    __s, ios_base & __io, char_type __fill, double __v ) const    [protected], [virtual]
```

Numeric formatting.

These functions do the work of formatting numeric values and inserting them into a stream. This function is a hook for derived classes to change the value returned.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

Definition at line 1158 of file locale_facets.tcc.

References `std::num_put<_CharT, _Outlter>::do_put()`.

5.897.4.7 `template<typename _CharT, typename _Outlter> _Outlter std::num_put<_CharT, _Outlter>::do_put (iter_type __s, ios_base & __io, char_type __fill, long double __v) const [protected], [virtual]`

Numeric formatting.

These functions do the work of formatting numeric values and inserting them into a stream. This function is a hook for derived classes to change the value returned.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

Definition at line 1172 of file locale_facets.tcc.

References `std::num_put<_CharT, _Outlter>::do_put()`.

5.897.4.8 `template<typename _CharT, typename _Outlter> _Outlter std::num_put<_CharT, _Outlter>::do_put (iter_type __s, ios_base & __io, char_type __fill, const void * __v) const [protected], [virtual]`

Numeric formatting.

These functions do the work of formatting numeric values and inserting them into a stream. This function is a hook for derived classes to change the value returned.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

Definition at line 1179 of file locale_facets.tcc.

References `std::ios_base::_M_getloc()`, `std::ios_base::adjustfield`, `std::ios_base::basefield`, `std::ios_base::flags()`, `std::ios_base::hex`, `std::ios_base::internal`, `std::ios_base::left`, `std::ios_base::showbase`, `std::ios_base::uppercase`, and `std::__ctype_abstract_base<_CharT>::widen()`.

5.897.4.9 `template<typename _CharT, typename _Outlter> iter_type std::num_put<_CharT, _Outlter>::put (iter_type __s, ios_base & __io, char_type __fill, bool __v) const [inline]`

Numeric formatting.

Formats the boolean `v` and inserts it into a stream. It does so by calling `num_put::do_put()`.

If `ios_base::boolalpha` is set, writes `ctype<CharT>::truename()` or `ctype<CharT>::falsename()`. Otherwise formats `v` as an int.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

Definition at line 2328 of file locale_facets.h.

Referenced by `std::basic_ostream<_CharT, _Traits>::sentry::sentry()`.

5.897.4.10 `template<typename _CharT, typename _Outlter> iter_type std::num_put<_CharT, _Outlter>::put (iter_type __s, ios_base & __io, char_type __fill, long __v) const [inline]`

Numeric formatting.

Formats the integral value `v` and inserts it into a stream. It does so by calling `num_put::do_put()`.

Formatting is affected by the flag settings in `io`.

The basic format is affected by the value of `io.flags()` & `ios_base::basefield`. If equal to `ios_base::oct`, formats like the printf `o` specifier. Else if equal to `ios_base::hex`, formats like `x` or `X` with `ios_base::uppercase` unset or set respectively.

Otherwise, formats like `d`, `ld`, `lld` for signed and `u`, `lu`, `llu` for unsigned values. Note that if both `oct` and `hex` are set, neither will take effect.

If `ios_base::showpos` is set, `'+'` is output before positive values. If `ios_base::showbase` is set, `'0'` precedes octal values (except 0) and `'0[xX]'` precedes hex values.

The decimal point character used is `numpunct::decimal_point()`. Thousands separators are inserted according to `numpunct::grouping()` and `numpunct::thousands_sep()`.

If `io.width()` is non-zero, enough *fill* characters are inserted to make the result at least that wide. If `(io.flags() & ios_base::adjustfield) == ios_base::left`, result is padded at the end. If `ios_base::internal`, then padding occurs immediately after either a `'+'` or `'-'` or after `'0x'` or `'0X'`. Otherwise, padding occurs at the beginning.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

Definition at line 2370 of file `locale_facets.h`.

```
5.897.4.11 template<typename _CharT, typename _Outiter> iter_type std::num_put<_CharT, _Outiter>::put ( iter_type
    __s, ios_base & __io, char_type __fill, unsigned long __v ) const [inline]
```

Numeric formatting.

Formats the integral value `v` and inserts it into a stream. It does so by calling `num_put::do_put()`.

Formatting is affected by the flag settings in `io`.

The basic format is affected by the value of `io.flags()` & `ios_base::basefield`. If equal to `ios_base::oct`, formats like the `printf o` specifier. Else if equal to `ios_base::hex`, formats like `x` or `X` with `ios_base::uppercase` unset or set respectively. Otherwise, formats like `d`, `ld`, `lld` for signed and `u`, `lu`, `llu` for unsigned values. Note that if both `oct` and `hex` are set, neither will take effect.

If `ios_base::showpos` is set, `'+'` is output before positive values. If `ios_base::showbase` is set, `'0'` precedes octal values (except 0) and `'0[xX]'` precedes hex values.

The decimal point character used is `numpunct::decimal_point()`. Thousands separators are inserted according to `numpunct::grouping()` and `numpunct::thousands_sep()`.

If `io.width()` is non-zero, enough *fill* characters are inserted to make the result at least that wide. If `(io.flags() & ios_base::adjustfield) == ios_base::left`, result is padded at the end. If `ios_base::internal`, then padding occurs immediately after either a `'+'` or `'-'` or after `'0x'` or `'0X'`. Otherwise, padding occurs at the beginning.

Parameters

\leftrightarrow _s	Stream to write to.
\leftrightarrow _io	Source of locale and flags.
\leftrightarrow _fill	Char_type to use for filling.
\leftrightarrow _v	Value to format and insert.

Returns

Iterator after writing.

Definition at line 2374 of file locale_facets.h.

5.897.4.12 `template<typename _CharT, typename _Outiter > iter_type std::num_put<_CharT, _Outiter >::put (iter_type __s, ios_base & __io, char_type __fill, long long __v) const [inline]`

Numeric formatting.

Formats the integral value *v* and inserts it into a stream. It does so by calling `num_put::do_put()`.

Formatting is affected by the flag settings in *io*.

The basic format is affected by the value of `io.flags()` & `ios_base::basefield`. If equal to `ios_base::oct`, formats like the `printf o` specifier. Else if equal to `ios_base::hex`, formats like `x` or `X` with `ios_base::uppercase` unset or set respectively. Otherwise, formats like `d`, `ld`, `lld` for signed and `u`, `lu`, `llu` for unsigned values. Note that if both `oct` and `hex` are set, neither will take effect.

If `ios_base::showpos` is set, '+' is output before positive values. If `ios_base::showbase` is set, '0' precedes octal values (except 0) and '0[xX]' precedes hex values.

The decimal point character used is `num_punct::decimal_point()`. Thousands separators are inserted according to `num_punct::grouping()` and `num_punct::thousands_sep()`.

If `io.width()` is non-zero, enough *fill* characters are inserted to make the result at least that wide. If `(io.flags() & ios_base::adjustfield) == ios_base::left`, result is padded at the end. If `ios_base::internal`, then padding occurs immediately after either a '+' or '-' or after '0x' or '0X'. Otherwise, padding occurs at the beginning.

Parameters

\leftrightarrow _s	Stream to write to.
\leftrightarrow _io	Source of locale and flags.
\leftrightarrow _fill	Char_type to use for filling.
\leftrightarrow _v	Value to format and insert.

Returns

Iterator after writing.

Definition at line 2380 of file locale_facets.h.

```
5.897.4.13  template<typename _CharT, typename _Outiter > iter_type std::num_put< _CharT, _Outiter >::put ( iter_type
            __s, ios_base & __io, char_type __fill, unsigned long long __v ) const    [inline]
```

Numeric formatting.

Formats the integral value *v* and inserts it into a stream. It does so by calling num_put::do_put().

Formatting is affected by the flag settings in *io*.





The basic format is affected by the value of io.flags() & ios_base::basefield. If equal to ios_base::oct, formats like the printf o specifier. Else if equal to ios_base::hex, formats like x or X with ios_base::uppercase unset or set respectively. Otherwise, formats like d, ld, lld for signed and u, lu, llu for unsigned values. Note that if both oct and hex are set, neither will take effect.

If ios_base::showpos is set, '+' is output before positive values. If ios_base::showbase is set, '0' precedes octal values (except 0) and '0[xX]' precedes hex values.

The decimal point character used is numpunct::decimal_point(). Thousands separators are inserted according to numpunct::grouping() and numpunct::thousands_sep().

If io.width() is non-zero, enough *fill* characters are inserted to make the result at least that wide. If (io.flags() & ios_base::adjustfield) == ios_base::left, result is padded at the end. If ios_base::internal, then padding occurs immediately after either a '+' or '-' or after '0x' or '0X'. Otherwise, padding occurs at the beginning.

Parameters

 <i>__s</i>	Stream to write to.
 <i>__io</i>	Source of locale and flags.
 <i>__fill</i>	Char_type to use for filling.
 <i>__v</i>	Value to format and insert.

Returns

Iterator after writing.

Definition at line 2384 of file locale_facets.h.

```
5.897.4.14  template<typename _CharT, typename _Outiter > iter_type std::num_put< _CharT, _Outiter >::put ( iter_type
            __s, ios_base & __io, char_type __fill, double __v ) const    [inline]
```

Numeric formatting.

Formats the floating point value *v* and inserts it into a stream. It does so by calling num_put::do_put().

Formatting is affected by the flag settings in *io*.

The basic format is affected by the value of io.flags() & ios_base::floatfield. If equal to ios_base::fixed, formats like the printf f specifier. Else if equal to ios_base::scientific, formats like e or E with ios_base::uppercase unset or set respectively. Otherwise, formats like g or G depending on uppercase. Note that if both fixed and scientific are set, the effect will also be like g or G.

The output precision is given by io.precision(). This precision is capped at numeric_limits::digits10 + 2 (different for double and long double). The default precision is 6.

If ios_base::showpos is set, '+' is output before positive values. If ios_base::showpoint is set, a decimal point will always be output.

The decimal point character used is numpunct::decimal_point(). Thousands separators are inserted according to numpunct::grouping() and numpunct::thousands_sep().

If io.width() is non-zero, enough *fill* characters are inserted to make the result at least that wide. If (io.flags() & ios_base::adjustfield) == ios_base::left, result is padded at the end. If ios_base::internal, then padding occurs immediately after either a '+' or '-' or after '0x' or '0X'. Otherwise, padding occurs at the beginning.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

Definition at line 2433 of file locale_facets.h.

```
5.897.4.15 template<typename _CharT, typename _OutIter> iter_type std::num_put<_CharT, _OutIter>::put ( iter_type
    __s, ios_base & __io, char_type __fill, long double __v ) const [inline]
```

Numeric formatting.

Formats the floating point value *v* and inserts it into a stream. It does so by calling num_put::do_put().

Formatting is affected by the flag settings in *io*.

The basic format is affected by the value of io.flags() & ios_base::floatfield. If equal to ios_base::fixed, formats like the printf f specifier. Else if equal to ios_base::scientific, formats like e or E with ios_base::uppercase unset or set

respectively. Otherwise, formats like g or G depending on uppercase. Note that if both fixed and scientific are set, the effect will also be like g or G.

The output precision is given by `io.precision()`. This precision is capped at `numeric_limits::digits10 + 2` (different for double and long double). The default precision is 6.

If `ios_base::showpos` is set, '+' is output before positive values. If `ios_base::showpoint` is set, a decimal point will always be output.

The decimal point character used is `num_punct::decimal_point()`. Thousands separators are inserted according to `num_punct::grouping()` and `num_punct::thousands_sep()`.

If `io.width()` is non-zero, enough *fill* characters are inserted to make the result at least that wide. If `(io.flags() & ios_base::adjustfield) == ios_base::left`, result is padded at the end. If `ios_base::internal`, then padding occurs immediately after either a '+' or '-' or after '0x' or '0X'. Otherwise, padding occurs at the beginning.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

Definition at line 2437 of file `locale_facets.h`.

```
5.897.4.16 template<typename _CharT, typename _OutIter> iter_type std::num_put<_CharT, _OutIter>::put ( iter_type
    __s, ios_base & __io, char_type __fill, const void * __v ) const [inline]
```

Numeric formatting.

Formats the pointer value `v` and inserts it into a stream. It does so by calling `num_put::do_put()`.

This function formats `v` as an unsigned long with `ios_base::hex` and `ios_base::showbase` set.

Parameters

<code>__s</code>	Stream to write to.
<code>__io</code>	Source of locale and flags.
<code>__fill</code>	Char_type to use for filling.
<code>__v</code>	Value to format and insert.

Returns

Iterator after writing.

Definition at line 2458 of file locale_facets.h.

5.897.5 Member Data Documentation

5.897.5.1 `template<typename _CharT, typename _OutIter> locale::id std::num_put<_CharT, _OutIter>::id` `[static]`

Numpunct facet id.

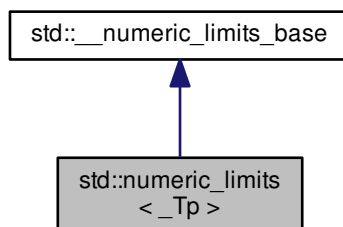
Definition at line 2300 of file locale_facets.h.

The documentation for this class was generated from the following files:

- [locale_facets.h](#)
- [locale_facets.tcc](#)

5.898 std::numeric_limits< _Tp > Struct Template Reference

Inheritance diagram for std::numeric_limits< _Tp >:

**Static Public Member Functions**

- static constexpr _Tp [denorm_min](#) () noexcept
- static constexpr _Tp [epsilon](#) () noexcept
- static constexpr _Tp [infinity](#) () noexcept
- static constexpr _Tp [lowest](#) () noexcept
- static constexpr _Tp [max](#) () noexcept
- static constexpr _Tp [min](#) () noexcept
- static constexpr _Tp [quiet_NaN](#) () noexcept
- static constexpr _Tp [round_error](#) () noexcept
- static constexpr _Tp [signaling_NaN](#) () noexcept

Static Public Attributes

- static constexpr int [digits](#)
- static constexpr int [digits10](#)
- static constexpr [float_denorm_style](#) [has_denorm](#)
- static constexpr bool [has_denorm_loss](#)
- static constexpr bool [has_infinity](#)
- static constexpr bool [has_quiet_NaN](#)
- static constexpr bool [has_signaling_NaN](#)
- static constexpr bool [is_bounded](#)
- static constexpr bool [is_exact](#)
- static constexpr bool [is_iec559](#)
- static constexpr bool [is_integer](#)
- static constexpr bool [is_modulo](#)
- static constexpr bool [is_signed](#)
- static constexpr bool [is_specialized](#)
- static constexpr int [max_digits10](#)
- static constexpr int [max_exponent](#)
- static constexpr int [max_exponent10](#)
- static constexpr int [min_exponent](#)
- static constexpr int [min_exponent10](#)
- static constexpr int [radix](#)
- static constexpr [float_round_style](#) [round_style](#)
- static constexpr bool [tinyness_before](#)
- static constexpr bool [traps](#)

5.898.1 Detailed Description

```
template<typename _Tp>
struct std::numeric_limits< _Tp >
```

Properties of fundamental types.

This class allows a program to obtain information about the representation of a fundamental type on a given platform. For non-fundamental types, the functions will return 0 and the data members will all be `false`.

`_GLIBCXX_RESOLVE_LIB_DEFECTS`: DRs 201 and 184 (hi Gaby!) are noted, but not incorporated in this documented (yet).

Definition at line 315 of file `limits`.

5.898.2 Member Function Documentation

5.898.2.1 `template<typename _Tp> static constexpr _Tp std::numeric_limits<_Tp>::denorm_min () [inline], [static], [noexcept]`

The minimum positive denormalized value. For types where `has_denorm` is `false`, this is the minimum positive normalized value.

Definition at line 360 of file `limits`.

5.898.2.2 `template<typename _Tp> static constexpr _Tp std::numeric_limits<_Tp>::epsilon () [inline],
[static], [noexcept]`

The *machine epsilon*: the difference between 1 and the least value greater than 1 that is representable.

Definition at line 336 of file limits.

Referenced by `std::binomial_distribution<_IntType>::operator()()`, `std::poisson_distribution<_IntType>::operator()()`, and `std::operator<<()`.

5.898.2.3 `template<typename _Tp> static constexpr _Tp std::numeric_limits<_Tp>::infinity () [inline],
[static], [noexcept]`

The representation of positive infinity, if `has_infinity`.

Definition at line 344 of file limits.

5.898.2.4 `template<typename _Tp> static constexpr _Tp std::numeric_limits<_Tp>::lowest () [inline],
[static], [noexcept]`

A finite value `x` such that there is no other finite value `y` where `y < x`.

Definition at line 330 of file limits.

Referenced by `std::normal_distribution<result_type>::min()`, `std::cauchy_distribution<_RealType>::min()`, `std::student_t_distribution<_RealType>::min()`, and `std::extreme_value_distribution<_RealType>::min()`.

5.898.2.5 `template<typename _Tp> static constexpr _Tp std::numeric_limits<_Tp>::max () [inline],
[static], [noexcept]`

The maximum finite value.

Definition at line 324 of file limits.

Referenced by `std::normal_distribution<result_type>::max()`, `std::lognormal_distribution<_RealType>::max()`, `std::gamma_distribution<result_type>::max()`, `std::chi_squared_distribution<_RealType>::max()`, `std::cauchy_distribution<_RealType>::max()`, `std::fisher_f_distribution<_RealType>::max()`, `std::student_t_distribution<_RealType>::max()`, `std::bernoulli_distribution::max()`, `std::geometric_distribution<_IntType>::max()`, `std::negative_binomial_distribution<_IntType>::max()`, `std::poisson_distribution<_IntType>::max()`, `std::exponential_distribution<_RealType>::max()`, `std::weibull_distribution<_RealType>::max()`, `std::extreme_value_distribution<_RealType>::max()`, `std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType>::operator()()`, `std::binomial_distribution<_IntType>::operator()()`, `std::poisson_distribution<_IntType>::operator()()`, and `std::operator<<()`.

5.898.2.6 `template<typename _Tp> static constexpr _Tp std::numeric_limits<_Tp>::min () [inline], [static],
[noexcept]`

The minimum finite value, or for floating types with denormalization, the minimum positive normalized value.

Definition at line 320 of file limits.

Referenced by `std::bernoulli_distribution::min()`, and `std::independent_bits_engine<_RandomNumberEngine, __w, _UIntType>::operator()()`.

5.898.2.7 `template<typename _Tp> static constexpr _Tp std::numeric_limits<_Tp>::quiet_NaN() [inline],
[static],[noexcept]`

The representation of a quiet Not a Number, if `has_quiet_NaN`.

Definition at line 349 of file `limits`.

5.898.2.8 `template<typename _Tp> static constexpr _Tp std::numeric_limits<_Tp>::round_error() [inline],
[static],[noexcept]`

The maximum rounding error measurement (see LIA-1).

Definition at line 340 of file `limits`.

5.898.2.9 `template<typename _Tp> static constexpr _Tp std::numeric_limits<_Tp>::signaling_NaN() [inline],
[static],[noexcept]`

The representation of a signaling Not a Number, if `has_signaling_NaN`.

Definition at line 354 of file `limits`.

5.898.3 Member Data Documentation

5.898.3.1 `constexpr int std::__numeric_limits_base::digits [static],[inherited]`

The number of `radix` digits that be represented without change: for integer types, the number of non-sign bits in the mantissa; for floating types, the number of `radix` digits in the mantissa.

Definition at line 211 of file `limits`.

5.898.3.2 `constexpr int std::__numeric_limits_base::digits10 [static],[inherited]`

The number of base 10 digits that can be represented without change.

Definition at line 214 of file `limits`.

5.898.3.3 `constexpr float_denorm_style std::__numeric_limits_base::has_denorm [static],[inherited]`

See `std::float_denorm_style` for more information.

Definition at line 266 of file `limits`.

5.898.3.4 `constexpr bool std::__numeric_limits_base::has_denorm_loss [static],[inherited]`

True if loss of accuracy is detected as a denormalization loss, rather than as an inexact result.

Definition at line 270 of file `limits`.

5.898.3.5 constexpr bool std::__numeric_limits_base::has_infinity [static],[inherited]

True if the type has a representation for positive infinity.

Definition at line 255 of file limits.

5.898.3.6 constexpr bool std::__numeric_limits_base::has_quiet_NaN [static],[inherited]

True if the type has a representation for a quiet (non-signaling) Not a Number.

Definition at line 259 of file limits.

5.898.3.7 constexpr bool std::__numeric_limits_base::has_signaling_NaN [static],[inherited]

True if the type has a representation for a signaling Not a Number.

Definition at line 263 of file limits.

5.898.3.8 constexpr bool std::__numeric_limits_base::is_bounded [static],[inherited]

True if the set of values representable by the type is finite. All built-in types are bounded, this member would be false for arbitrary precision types.

Definition at line 279 of file limits.

5.898.3.9 constexpr bool std::__numeric_limits_base::is_exact [static],[inherited]

True if the type uses an exact representation. All integer types are exact, but not all exact types are integer. For example, rational and fixed-exponent representations are exact but not integer.

Definition at line 231 of file limits.

5.898.3.10 constexpr bool std::__numeric_limits_base::is_iec559 [static],[inherited]

True if-and-only-if the type adheres to the IEC 559 standard, also known as IEEE 754. (Only makes sense for floating point types.)

Definition at line 274 of file limits.

5.898.3.11 constexpr bool std::__numeric_limits_base::is_integer [static],[inherited]

True if the type is integer.

Definition at line 226 of file limits.

5.898.3.12 `constexpr bool std::__numeric_limits_base::is_modulo` `[static],[inherited]`

True if the type is *modulo*. A type is modulo if, for any operation involving `+`, `-`, or `*` on values of that type whose result would fall outside the range `[min(),max()]`, the value returned differs from the true value by an integer multiple of `max() - min() + 1`. On most machines, this is false for floating types, true for unsigned integers, and true for signed integers. See PR22200 about signed integers.

Definition at line 288 of file `limits`.

5.898.3.13 `constexpr bool std::__numeric_limits_base::is_signed` `[static],[inherited]`

True if the type is signed.

Definition at line 223 of file `limits`.

5.898.3.14 `constexpr bool std::__numeric_limits_base::is_specialized` `[static],[inherited]`

This will be true for all fundamental types (which have specializations), and false for everything else.

Definition at line 206 of file `limits`.

5.898.3.15 `constexpr int std::__numeric_limits_base::max_digits10` `[static],[inherited]`

The number of base 10 digits required to ensure that values which differ are always differentiated.

Definition at line 219 of file `limits`.

5.898.3.16 `constexpr int std::__numeric_limits_base::max_exponent` `[static],[inherited]`

The maximum positive integer such that `radix` raised to the power of (one less than that integer) is a representable finite floating point number.

Definition at line 248 of file `limits`.

5.898.3.17 `constexpr int std::__numeric_limits_base::max_exponent10` `[static],[inherited]`

The maximum positive integer such that 10 raised to that power is in the range of representable finite floating point numbers.

Definition at line 252 of file `limits`.

5.898.3.18 `constexpr int std::__numeric_limits_base::min_exponent` `[static],[inherited]`

The minimum negative integer such that `radix` raised to the power of (one less than that integer) is a normalized floating point number.

Definition at line 239 of file `limits`.

5.898.3.19 `constexpr int std::__numeric_limits_base::min_exponent10` `[static], [inherited]`

The minimum negative integer such that 10 raised to that power is in the range of normalized floating point numbers.

Definition at line 243 of file `limits`.

5.898.3.20 `constexpr int std::__numeric_limits_base::radix` `[static], [inherited]`

For integer types, specifies the base of the representation. For floating types, specifies the base of the exponent representation.

Definition at line 235 of file `limits`.

5.898.3.21 `constexpr float_round_style std::__numeric_limits_base::round_style` `[static], [inherited]`

See `std::float_round_style` for more information. This is only meaningful for floating types; integer types will all be `round_toward_zero`.

Definition at line 299 of file `limits`.

5.898.3.22 `constexpr bool std::__numeric_limits_base::tinyness_before` `[static], [inherited]`

True if tininess is detected before rounding. (see IEC 559)

Definition at line 294 of file `limits`.

5.898.3.23 `constexpr bool std::__numeric_limits_base::traps` `[static], [inherited]`

True if trapping is implemented for this type.

Definition at line 291 of file `limits`.

The documentation for this struct was generated from the following file:

- [limits](#)

5.899 `std::numeric_limits< bool >` Struct Template Reference

Static Public Member Functions

- static `constexpr bool` **denorm_min** () noexcept
- static `constexpr bool` **epsilon** () noexcept
- static `constexpr bool` **infinity** () noexcept
- static `constexpr bool` **lowest** () noexcept
- static `constexpr bool` **max** () noexcept
- static `constexpr bool` **min** () noexcept
- static `constexpr bool` **quiet_NaN** () noexcept
- static `constexpr bool` **round_error** () noexcept
- static `constexpr bool` **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

5.899.1 Detailed Description

```
template<>
struct std::numeric_limits< bool >
```

numeric_limits<bool> specialization.

Definition at line 382 of file limits.

The documentation for this struct was generated from the following file:

- [limits](#)

5.900 std::numeric_limits< char > Struct Template Reference

Static Public Member Functions

- static constexpr char **denorm_min** () noexcept
- static constexpr char **epsilon** () noexcept
- static constexpr char **infinity** () noexcept
- static constexpr char **lowest** () noexcept
- static constexpr char **max** () noexcept
- static constexpr char **min** () noexcept
- static constexpr char **quiet_NaN** () noexcept
- static constexpr char **round_error** () noexcept
- static constexpr char **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

5.900.1 Detailed Description

```
template<>
struct std::numeric_limits< char >
```

`numeric_limits<char>` specialization.

Definition at line 451 of file `limits`.

The documentation for this struct was generated from the following file:

- [limits](#)

5.901 `std::numeric_limits< char16_t >` Struct Template Reference

Static Public Member Functions

- static constexpr `char16_t` **denorm_min** () noexcept
- static constexpr `char16_t` **epsilon** () noexcept
- static constexpr `char16_t` **infinity** () noexcept
- static constexpr `char16_t` **lowest** () noexcept
- static constexpr `char16_t` **max** () noexcept
- static constexpr `char16_t` **min** () noexcept
- static constexpr `char16_t` **quiet_NaN** () noexcept
- static constexpr `char16_t` **round_error** () noexcept
- static constexpr `char16_t` **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

5.901.1 Detailed Description

```
template<>
struct std::numeric_limits< char16_t >
```

numeric_limits<char16_t> specialization.

Definition at line 730 of file limits.

The documentation for this struct was generated from the following file:

- [limits](#)

5.902 std::numeric_limits< char32_t > Struct Template Reference

Static Public Member Functions

- static constexpr char32_t **denorm_min** () noexcept
- static constexpr char32_t **epsilon** () noexcept
- static constexpr char32_t **infinity** () noexcept
- static constexpr char32_t **lowest** () noexcept
- static constexpr char32_t **max** () noexcept
- static constexpr char32_t **min** () noexcept
- static constexpr char32_t **quiet_NaN** () noexcept
- static constexpr char32_t **round_error** () noexcept
- static constexpr char32_t **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

5.902.1 Detailed Description

```
template<>
struct std::numeric_limits< char32_t >
```

`numeric_limits<char32_t>` specialization.

Definition at line 791 of file `limits`.

The documentation for this struct was generated from the following file:

- [limits](#)

5.903 `std::numeric_limits< double >` Struct Template Reference

Static Public Member Functions

- static constexpr double **denorm_min** () noexcept
- static constexpr double **epsilon** () noexcept
- static constexpr double **infinity** () noexcept
- static constexpr double **lowest** () noexcept
- static constexpr double **max** () noexcept
- static constexpr double **min** () noexcept
- static constexpr double **quiet_NaN** () noexcept
- static constexpr double **round_error** () noexcept
- static constexpr double **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

5.903.1 Detailed Description

```
template<>
struct std::numeric_limits< double >
```

numeric_limits<double> specialization.

Definition at line 1668 of file limits.

The documentation for this struct was generated from the following file:

- [limits](#)

5.904 std::numeric_limits< float > Struct Template Reference

Static Public Member Functions

- static constexpr float **denorm_min** () noexcept
- static constexpr float **epsilon** () noexcept
- static constexpr float **infinity** () noexcept
- static constexpr float **lowest** () noexcept
- static constexpr float **max** () noexcept
- static constexpr float **min** () noexcept
- static constexpr float **quiet_NaN** () noexcept
- static constexpr float **round_error** () noexcept
- static constexpr float **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

5.904.1 Detailed Description

```
template<>
struct std::numeric_limits< float >
```

`numeric_limits<float>` specialization.

Definition at line 1593 of file `limits`.

The documentation for this struct was generated from the following file:

- [limits](#)

5.905 `std::numeric_limits< int >` Struct Template Reference

Static Public Member Functions

- static constexpr int **denorm_min** () noexcept
- static constexpr int **epsilon** () noexcept
- static constexpr int **infinity** () noexcept
- static constexpr int **lowest** () noexcept
- static constexpr int **max** () noexcept
- static constexpr int **min** () noexcept
- static constexpr int **quiet_NaN** () noexcept
- static constexpr int **round_error** () noexcept
- static constexpr int **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

5.905.1 Detailed Description

```
template<>
struct std::numeric_limits< int >
```

numeric_limits<int> specialization.

Definition at line 993 of file limits.

The documentation for this struct was generated from the following file:

- [limits](#)

5.906 std::numeric_limits< long > Struct Template Reference

Static Public Member Functions

- static constexpr long **denorm_min** () noexcept
- static constexpr long **epsilon** () noexcept
- static constexpr long **infinity** () noexcept
- static constexpr long **lowest** () noexcept
- static constexpr long **max** () noexcept
- static constexpr long **min** () noexcept
- static constexpr long **quiet_NaN** () noexcept
- static constexpr long **round_error** () noexcept
- static constexpr long **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

5.906.1 Detailed Description

```
template<>
struct std::numeric_limits< long >
```

`numeric_limits<long>` specialization.

Definition at line 1132 of file `limits`.

The documentation for this struct was generated from the following file:

- [limits](#)

5.907 `std::numeric_limits< long double >` Struct Template Reference

Static Public Member Functions

- static constexpr long double **denorm_min** () noexcept
- static constexpr long double **epsilon** () noexcept
- static constexpr long double **infinity** () noexcept
- static constexpr long double **lowest** () noexcept
- static constexpr long double **max** () noexcept
- static constexpr long double **min** () noexcept
- static constexpr long double **quiet_NaN** () noexcept
- static constexpr long double **round_error** () noexcept
- static constexpr long double **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

5.907.1 Detailed Description

```
template<>
struct std::numeric_limits< long double >
```

numeric_limits<long double> specialization.

Definition at line 1743 of file limits.

The documentation for this struct was generated from the following file:

- [limits](#)

5.908 std::numeric_limits< long long > Struct Template Reference

Static Public Member Functions

- static constexpr long long **denorm_min** () noexcept
- static constexpr long long **epsilon** () noexcept
- static constexpr long long **infinity** () noexcept
- static constexpr long long **lowest** () noexcept
- static constexpr long long **max** () noexcept
- static constexpr long long **min** () noexcept
- static constexpr long long **quiet_NaN** () noexcept
- static constexpr long long **round_error** () noexcept
- static constexpr long long **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

5.908.1 Detailed Description

```
template<>
struct std::numeric_limits< long long >
```

`numeric_limits<long long>` specialization.

Definition at line 1272 of file `limits`.

The documentation for this struct was generated from the following file:

- [limits](#)

5.909 `std::numeric_limits< short >` Struct Template Reference

Static Public Member Functions

- static constexpr short **denorm_min** () noexcept
- static constexpr short **epsilon** () noexcept
- static constexpr short **infinity** () noexcept
- static constexpr short **lowest** () noexcept
- static constexpr short **max** () noexcept
- static constexpr short **min** () noexcept
- static constexpr short **quiet_NaN** () noexcept
- static constexpr short **round_error** () noexcept
- static constexpr short **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

5.909.1 Detailed Description

```
template<>
struct std::numeric_limits< short >
```

numeric_limits<short> specialization.

Definition at line 853 of file limits.

The documentation for this struct was generated from the following file:

- [limits](#)

5.910 std::numeric_limits< signed char > Struct Template Reference

Static Public Member Functions

- static constexpr signed char **denorm_min** () noexcept
- static constexpr signed char **epsilon** () noexcept
- static constexpr signed char **infinity** () noexcept
- static constexpr signed char **lowest** () noexcept
- static constexpr signed char **max** () noexcept
- static constexpr signed char **min** () noexcept
- static constexpr signed char **quiet_NaN** () noexcept
- static constexpr signed char **round_error** () noexcept
- static constexpr signed char **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

5.910.1 Detailed Description

```
template<>
```

```
struct std::numeric_limits< signed char >
```

`numeric_limits<signed char>` specialization.

Definition at line 518 of file `limits`.

The documentation for this struct was generated from the following file:

- [limits](#)

5.911 `std::numeric_limits< unsigned char >` Struct Template Reference

Static Public Member Functions

- static constexpr unsigned char **denorm_min** () noexcept
- static constexpr unsigned char **epsilon** () noexcept
- static constexpr unsigned char **infinity** () noexcept
- static constexpr unsigned char **lowest** () noexcept
- static constexpr unsigned char **max** () noexcept
- static constexpr unsigned char **min** () noexcept
- static constexpr unsigned char **quiet_NaN** () noexcept
- static constexpr unsigned char **round_error** () noexcept
- static constexpr unsigned char **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

5.911.1 Detailed Description

```
template<>
struct std::numeric_limits< unsigned char >
```

numeric_limits<unsigned char> specialization.

Definition at line 588 of file limits.

The documentation for this struct was generated from the following file:

- [limits](#)

5.912 std::numeric_limits< unsigned int > Struct Template Reference

Static Public Member Functions

- static constexpr unsigned int **denorm_min** () noexcept
- static constexpr unsigned int **epsilon** () noexcept
- static constexpr unsigned int **infinity** () noexcept
- static constexpr unsigned int **lowest** () noexcept
- static constexpr unsigned int **max** () noexcept
- static constexpr unsigned int **min** () noexcept
- static constexpr unsigned int **quiet_NaN** () noexcept
- static constexpr unsigned int **round_error** () noexcept
- static constexpr unsigned int **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

5.912.1 Detailed Description

```
template<>
```

```
struct std::numeric_limits< unsigned int >
```

`numeric_limits<unsigned int>` specialization.

Definition at line 1060 of file `limits`.

The documentation for this struct was generated from the following file:

- [limits](#)

5.913 `std::numeric_limits< unsigned long >` Struct Template Reference

Static Public Member Functions

- static constexpr unsigned long **denorm_min** () noexcept
- static constexpr unsigned long **epsilon** () noexcept
- static constexpr unsigned long **infinity** () noexcept
- static constexpr unsigned long **lowest** () noexcept
- static constexpr unsigned long **max** () noexcept
- static constexpr unsigned long **min** () noexcept
- static constexpr unsigned long **quiet_NaN** () noexcept
- static constexpr unsigned long **round_error** () noexcept
- static constexpr unsigned long **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

5.913.1 Detailed Description

```
template<>
struct std::numeric_limits< unsigned long >
```

numeric_limits<unsigned long> specialization.

Definition at line 1199 of file limits.

The documentation for this struct was generated from the following file:

- [limits](#)

5.914 std::numeric_limits< unsigned long long > Struct Template Reference

Static Public Member Functions

- static constexpr unsigned long long **denorm_min** () noexcept
- static constexpr unsigned long long **epsilon** () noexcept
- static constexpr unsigned long long **infinity** () noexcept
- static constexpr unsigned long long **lowest** () noexcept
- static constexpr unsigned long long **max** () noexcept
- static constexpr unsigned long long **min** () noexcept
- static constexpr unsigned long long **quiet_NaN** () noexcept
- static constexpr unsigned long long **round_error** () noexcept
- static constexpr unsigned long long **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

5.914.1 Detailed Description

```
template<>
struct std::numeric_limits< unsigned long long >
```

`numeric_limits<unsigned long long>` specialization.

Definition at line 1342 of file `limits`.

The documentation for this struct was generated from the following file:

- [limits](#)

5.915 `std::numeric_limits< unsigned short >` Struct Template Reference

Static Public Member Functions

- static constexpr unsigned short **denorm_min** () noexcept
- static constexpr unsigned short **epsilon** () noexcept
- static constexpr unsigned short **infinity** () noexcept
- static constexpr unsigned short **lowest** () noexcept
- static constexpr unsigned short **max** () noexcept
- static constexpr unsigned short **min** () noexcept
- static constexpr unsigned short **quiet_NaN** () noexcept
- static constexpr unsigned short **round_error** () noexcept
- static constexpr unsigned short **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

5.915.1 Detailed Description

```
template<>
struct std::numeric_limits< unsigned short >
```

numeric_limits<unsigned short> specialization.

Definition at line 920 of file limits.

The documentation for this struct was generated from the following file:

- [limits](#)

5.916 std::numeric_limits< wchar_t > Struct Template Reference

Static Public Member Functions

- static constexpr wchar_t **denorm_min** () noexcept
- static constexpr wchar_t **epsilon** () noexcept
- static constexpr wchar_t **infinity** () noexcept
- static constexpr wchar_t **lowest** () noexcept
- static constexpr wchar_t **max** () noexcept
- static constexpr wchar_t **min** () noexcept
- static constexpr wchar_t **quiet_NaN** () noexcept
- static constexpr wchar_t **round_error** () noexcept
- static constexpr wchar_t **signaling_NaN** () noexcept

Static Public Attributes

- static constexpr int **digits**
- static constexpr int **digits10**
- static constexpr [float_denorm_style](#) **has_denorm**
- static constexpr bool **has_denorm_loss**
- static constexpr bool **has_infinity**
- static constexpr bool **has_quiet_NaN**
- static constexpr bool **has_signaling_NaN**
- static constexpr bool **is_bounded**
- static constexpr bool **is_exact**
- static constexpr bool **is_iec559**
- static constexpr bool **is_integer**
- static constexpr bool **is_modulo**
- static constexpr bool **is_signed**
- static constexpr bool **is_specialized**
- static constexpr int **max_digits10**
- static constexpr int **max_exponent**
- static constexpr int **max_exponent10**
- static constexpr int **min_exponent**
- static constexpr int **min_exponent10**
- static constexpr int **radix**
- static constexpr [float_round_style](#) **round_style**
- static constexpr bool **tinyness_before**
- static constexpr bool **traps**

5.916.1 Detailed Description

```
template<>
struct std::numeric_limits< wchar_t >
```

`numeric_limits<wchar_t>` specialization.

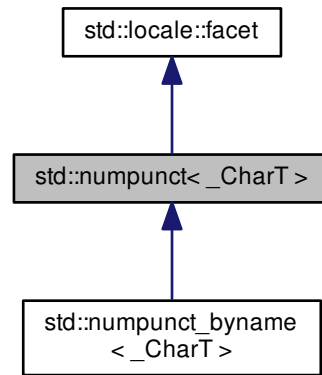
Definition at line 661 of file `limits`.

The documentation for this struct was generated from the following file:

- [limits](#)

5.917 `std::numpunct<_CharT>` Class Template Reference

Inheritance diagram for `std::numpunct<_CharT>`:



Public Types

- `typedef __numpunct_cache<_CharT> __cache_type`
- `typedef _CharT char_type`
- `typedef basic_string<_CharT> string_type`

Public Member Functions

- `numpunct` (`size_t __refs=0`)
- `numpunct` (`__cache_type *__cache, size_t __refs=0`)
- `numpunct` (`__c_locale __cloc, size_t __refs=0`)
- `char_type decimal_point` () const
- `string_type falsename` () const
- `string grouping` () const
- `char_type thousands_sep` () const
- `string_type truename` () const

Static Public Attributes

- static `locale::id id`

Protected Member Functions

- virtual `~numpunct()`
- void `_M_initialize_numpunct(__c_locale __cloc=0)`
- template<>
void `_M_initialize_numpunct(__c_locale __cloc)`
- template<>
void `_M_initialize_numpunct(__c_locale __cloc)`
- virtual `char_type do_decimal_point()` const
- virtual `string_type do_falsename()` const
- virtual `string do_grouping()` const
- virtual `char_type do_thousands_sep()` const
- virtual `string_type do_truename()` const

Static Protected Member Functions

- static `__c_locale _S_clone_c_locale(__c_locale &__cloc) throw()`
- static void `_S_create_c_locale(__c_locale &__cloc, const char *__s, __c_locale __old=0)`
- static void `_S_destroy_c_locale(__c_locale &__cloc)`
- static `__c_locale _S_get_c_locale()`
- static const char * `_S_get_c_name()` throw()
- static `__c_locale _S_lc_ctype_c_locale(__c_locale __cloc, const char *__s)`

Protected Attributes

- `__cache_type * _M_data`

5.917.1 Detailed Description

```
template<typename _CharT>
class std::numpunct<_CharT>
```

Primary class template numpunct.

This facet stores several pieces of information related to printing and scanning numbers, such as the decimal point character. It takes a template parameter specifying the char type. The numpunct facet is used by streams for many I/O operations involving numbers.

The numpunct template uses protected virtual functions to provide the actual results. The public accessors forward the call to the virtual functions. These virtual functions are hooks for developers to implement the behavior they require from a numpunct facet.

Definition at line 1666 of file locale_facets.h.

5.917.2 Member Typedef Documentation

5.917.2.1 `template<typename _CharT> typedef _CharT std::num_punct<_CharT>::char_type`

Public typedefs.

Definition at line 1672 of file locale_facets.h.

5.917.2.2 `template<typename _CharT> typedef basic_string<_CharT> std::num_punct<_CharT>::string_type`

Public typedefs.

Definition at line 1673 of file locale_facets.h.

5.917.3 Constructor & Destructor Documentation

5.917.3.1 `template<typename _CharT> std::num_punct<_CharT>::num_punct (size_t __refs = 0) [inline], [explicit]`

Num_punct constructor.

Parameters

<code>__refs</code>	Refcount to pass to the base class.
---------------------	-------------------------------------

Definition at line 1690 of file locale_facets.h.

5.917.3.2 `template<typename _CharT> std::num_punct<_CharT>::num_punct (__cache_type * __cache, size_t __refs = 0) [inline], [explicit]`

Internal constructor. Not for general use.

This is a constructor for use by the library itself to set up the predefined locale facets.

Parameters

<code>__cache</code>	<code>__num_punct_cache</code> object.
<code>__refs</code>	Refcount to pass to the base class.

Definition at line 1704 of file locale_facets.h.

5.917.3.3 `template<typename _CharT> std::num_punct<_CharT>::num_punct (__c_locale __cloc, size_t __refs = 0) [inline], [explicit]`

Internal constructor. Not for general use.

This is a constructor for use by the library itself to set up new locales.

Parameters

<code>__cloc</code>	The C locale.
<code>__refs</code>	Refcount to pass to the base class.

Definition at line 1718 of file locale_facets.h.

5.917.3.4 `template<typename _CharT> virtual std::num_punct<_CharT>::~~num_punct ()` `[protected]`,
`[virtual]`

Destructor.

Referenced by `std::num_punct<_CharT>::do_falsename()`.

5.917.4 Member Function Documentation

5.917.4.1 `template<typename _CharT> char_type std::num_punct<_CharT>::decimal_point () const` `[inline]`

Return decimal point character.

This function returns a `char_type` to use as a decimal point. It does so by returning `num_punct<char_type>::do_decimal_point()`.

Returns

char_type representing a decimal point.

Definition at line 1732 of file locale_facets.h.

5.917.4.2 `template<typename _CharT> virtual char_type std::num_punct<_CharT>::do_decimal_point () const`
`[inline]`, `[protected]`, `[virtual]`

Return decimal point character.

Returns a `char_type` to use as a decimal point. This function is a hook for derived classes to change the value returned.

Returns

char_type representing a decimal point.

Definition at line 1819 of file locale_facets.h.

5.917.4.3 `template<typename _CharT> virtual string_type std::numpunct<_CharT>::do_falsename () const`
`[inline], [protected], [virtual]`

Return string representation of bool false.

Returns a `string_type` containing the text representation for false bool variables. This function is a hook for derived classes to change the value returned.

Returns

`string_type` representing printed form of false.

Definition at line 1870 of file `locale_facets.h`.

References `std::numpunct<_CharT>::~~numpunct()`.

5.917.4.4 `template<typename _CharT> virtual string std::numpunct<_CharT>::do_grouping () const` `[inline],`
`[protected], [virtual]`

Return grouping specification.

Returns a string representing groupings for the integer part of a number. This function is a hook for derived classes to change the value returned.

See also

`grouping()` for details.

Returns

String representing grouping specification.

Definition at line 1844 of file `locale_facets.h`.

5.917.4.5 `template<typename _CharT> virtual char_type std::numpunct<_CharT>::do_thousands_sep () const`
`[inline], [protected], [virtual]`

Return thousands separator character.

Returns a `char_type` to use as a thousands separator. This function is a hook for derived classes to change the value returned.

Returns

`char_type` representing a thousands separator.

Definition at line 1831 of file `locale_facets.h`.

5.917.4.6 `template<typename _CharT> virtual string_type std::num_punct<_CharT>::do_truename () const`
`[inline], [protected], [virtual]`

Return string representation of bool true.

Returns a string_type containing the text representation for true bool variables. This function is a hook for derived classes to change the value returned.

Returns

string_type representing printed form of true.

Definition at line 1857 of file locale_facets.h.

5.917.4.7 `template<typename _CharT> string_type std::num_punct<_CharT>::falsename () const` `[inline]`

Return string representation of bool false.

This function returns a string_type containing the text representation for false bool variables. It does so by calling num_punct<char_type>::do_falsename().

Returns

string_type representing printed form of false.

Definition at line 1802 of file locale_facets.h.

5.917.4.8 `template<typename _CharT> string std::num_punct<_CharT>::grouping () const` `[inline]`

Return grouping specification.

This function returns a string representing groupings for the integer part of a number. Groupings indicate where thousands separators should be inserted in the integer part of a number.

Each char in the return string is interpreted as an integer rather than a character. These numbers represent the number of digits in a group. The first char in the string represents the number of digits in the least significant group. If a char is negative, it indicates an unlimited number of digits for the group. If more chars from the string are required to group a number, the last char is used repeatedly.

For example, if the grouping() returns "\003\002" and is applied to the number 123456789, this corresponds to 12,34,56,789. Note that if the string was "32", this would put more than 50 digits into the least significant group if the character set is ASCII.

The string is returned by calling num_punct<char_type>::do_grouping().

Returns

string representing grouping specification.

Definition at line 1776 of file locale_facets.h.

5.917.4.9 `template<typename _CharT> char_type std::numpunct<_CharT>::thousands_sep () const [inline]`

Return thousands separator character.

This function returns a `char_type` to use as a thousands separator. It does so by returning `numpunct<char_type>::do_thousands_sep()`.

Returns

`char_type` representing a thousands separator.

Definition at line 1745 of file `locale_facets.h`.

5.917.4.10 `template<typename _CharT> string_type std::numpunct<_CharT>::truename () const [inline]`

Return string representation of `bool true`.

This function returns a `string_type` containing the text representation for `true` `bool` variables. It does so by calling `numpunct<char_type>::do_truename()`.

Returns

`string_type` representing printed form of `true`.

Definition at line 1789 of file `locale_facets.h`.

5.917.5 Member Data Documentation

5.917.5.1 `template<typename _CharT> locale::id std::numpunct<_CharT>::id [static]`

Numpunct facet id.

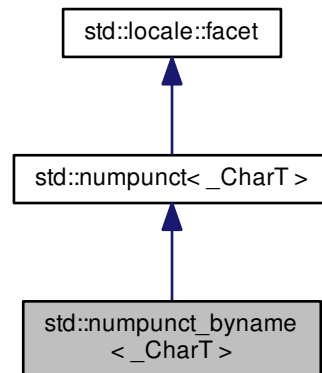
Definition at line 1682 of file `locale_facets.h`.

The documentation for this class was generated from the following file:

- [locale_facets.h](#)

5.918 std::numpunct_byname<_CharT> Class Template Reference

Inheritance diagram for std::numpunct_byname<_CharT>:



Public Types

- typedef __numpunct_cache<_CharT> **__cache_type**
- typedef _CharT **char_type**
- typedef [basic_string](#)<_CharT> **string_type**

Public Member Functions

- **numpunct_byname** (const char *__s, size_t __refs=0)
- **numpunct_byname** (const [string](#) &__s, size_t __refs=0)
- [char_type decimal_point](#) () const
- [string_type falsename](#) () const
- [string grouping](#) () const
- [char_type thousands_sep](#) () const
- [string_type truenam](#) () const

Static Public Attributes

- static [locale::id](#) id

Protected Member Functions

- void **_M_initialize_numpunct** (__c_locale __cloc=0)
- template<>
void **_M_initialize_numpunct** (__c_locale __cloc)
- template<>
void **_M_initialize_numpunct** (__c_locale __cloc)
- virtual [char_type do_decimal_point](#) () const
- virtual [string_type do_falsename](#) () const
- virtual [string do_grouping](#) () const
- virtual [char_type do_thousands_sep](#) () const
- virtual [string_type do_truename](#) () const

Static Protected Member Functions

- static __c_locale **_S_clone_c_locale** (__c_locale &__cloc) throw ()
- static void **_S_create_c_locale** (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale &__cloc)
- static __c_locale **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static __c_locale **_S_lc_ctype_c_locale** (__c_locale __cloc, const char *__s)

Protected Attributes

- __cache_type * **_M_data**

5.918.1 Detailed Description

```
template<typename _CharT>
class std::numpunct_byname< _CharT >
```

class numpunct_byname [22.2.3.2].

Definition at line 1899 of file locale_facets.h.

5.918.2 Member Function Documentation

5.918.2.1 `template<typename _CharT> char_type std::numpunct<_CharT>::decimal_point () const` `[inline]`,
`[inherited]`

Return decimal point character.

This function returns a `char_type` to use as a decimal point. It does so by returning `numpunct<char_type>::do_decimal_point()`.

Returns

char_type representing a decimal point.

Definition at line 1732 of file locale_facets.h.

5.918.2.2 `template<typename _CharT> virtual char_type std::num_punct<_CharT>::do_decimal_point () const`
[inline], [protected], [virtual], [inherited]

Return decimal point character.

Returns a `char_type` to use as a decimal point. This function is a hook for derived classes to change the value returned.

Returns

char_type representing a decimal point.

Definition at line 1819 of file `locale_facets.h`.

5.918.2.3 `template<typename _CharT> virtual string_type std::num_punct<_CharT>::do_falsename () const`
[inline], [protected], [virtual], [inherited]

Return string representation of bool false.

Returns a `string_type` containing the text representation for false bool variables. This function is a hook for derived classes to change the value returned.

Returns

string_type representing printed form of false.

Definition at line 1870 of file `locale_facets.h`.

References `std::num_punct<_CharT>::~num_punct()`.

5.918.2.4 `template<typename _CharT> virtual string std::num_punct<_CharT>::do_grouping () const` [inline],
[protected], [virtual], [inherited]

Return grouping specification.

Returns a string representing groupings for the integer part of a number. This function is a hook for derived classes to change the value returned.

See also

`grouping()` for details.

Returns

String representing grouping specification.

Definition at line 1844 of file `locale_facets.h`.

5.918.2.5 `template<typename _CharT> virtual char_type std::num_punct<_CharT>::do_thousands_sep () const`
`[inline], [protected], [virtual], [inherited]`

Return thousands separator character.

Returns a `char_type` to use as a thousands separator. This function is a hook for derived classes to change the value returned.

Returns

char_type representing a thousands separator.

Definition at line 1831 of file `locale_facets.h`.

5.918.2.6 `template<typename _CharT> virtual string_type std::num_punct<_CharT>::do_truename () const`
`[inline], [protected], [virtual], [inherited]`

Return string representation of bool true.

Returns a `string_type` containing the text representation for true bool variables. This function is a hook for derived classes to change the value returned.

Returns

`string_type` representing printed form of true.

Definition at line 1857 of file `locale_facets.h`.

5.918.2.7 `template<typename _CharT> string_type std::num_punct<_CharT>::falsename () const` `[inline],`
`[inherited]`

Return string representation of bool false.

This function returns a `string_type` containing the text representation for false bool variables. It does so by calling `num_punct<char_type>::do_falsename()`.

Returns

`string_type` representing printed form of false.

Definition at line 1802 of file `locale_facets.h`.

5.918.2.8 `template<typename _CharT> string std::num_punct<_CharT>::grouping () const [inline],
[inherited]`

Return grouping specification.

This function returns a string representing groupings for the integer part of a number. Groupings indicate where thousands separators should be inserted in the integer part of a number.

Each char in the return string is interpreted as an integer rather than a character. These numbers represent the number of digits in a group. The first char in the string represents the number of digits in the least significant group. If a char is negative, it indicates an unlimited number of digits for the group. If more chars from the string are required to group a number, the last char is used repeatedly.

For example, if the grouping() returns "\003\002" and is applied to the number 123456789, this corresponds to 12,34,56,789. Note that if the string was "32", this would put more than 50 digits into the least significant group if the character set is ASCII.

The string is returned by calling num_punct<char_type>::do_grouping().

Returns

string representing grouping specification.

Definition at line 1776 of file locale_facets.h.

5.918.2.9 `template<typename _CharT> char_type std::num_punct<_CharT>::thousands_sep () const [inline],
[inherited]`

Return thousands separator character.

This function returns a char_type to use as a thousands separator. It does so by returning num_punct<char_type>::do_thousands_sep().

Returns

char_type representing a thousands separator.

Definition at line 1745 of file locale_facets.h.

5.918.2.10 `template<typename _CharT> string_type std::num_punct<_CharT>::truename () const [inline],
[inherited]`

Return string representation of bool true.

This function returns a string_type containing the text representation for true bool variables. It does so by calling num_punct<char_type>::do_truename().

Returns

string_type representing printed form of true.

Definition at line 1789 of file locale_facets.h.

5.918.3 Member Data Documentation

5.918.3.1 `template<typename _CharT> locale::id std::numpunct<_CharT>::id` `[static], [inherited]`

Numpunct facet id.

Definition at line 1682 of file locale_facets.h.

The documentation for this class was generated from the following file:

- [locale_facets.h](#)

5.919 std::once_flag Struct Reference

Public Member Functions

- constexpr [once_flag](#) () noexcept=default
- [once_flag](#) (const [once_flag](#) &)=delete
- [once_flag](#) & operator= (const [once_flag](#) &)=delete

Friends

- template<typename _Callable, typename... _Args>
void [call_once](#) ([once_flag](#) &__once, _Callable &&__f, _Args &&...__args)

5.919.1 Detailed Description

[once_flag](#)

Definition at line 559 of file mutex.

5.919.2 Constructor & Destructor Documentation

5.919.2.1 `constexpr std::once_flag::once_flag ()` `[default], [noexcept]`

Constructor.

5.919.2.2 `std::once_flag::once_flag (const once_flag &)` `[delete]`

Deleted copy constructor.

5.919.3 Member Function Documentation

5.919.3.1 `once_flag& std::once_flag::operator=(const once_flag &)` `[delete]`

Deleted assignment operator.

5.919.4 Friends And Related Function Documentation

5.919.4.1 `template<typename _Callable, typename... _Args> void call_once (once_flag & __once, _Callable && __f, _Args &&... __args)` `[friend]`

`call_once`

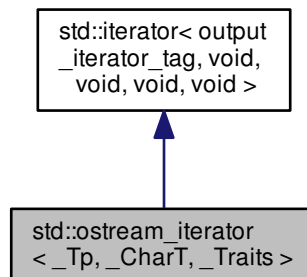
Definition at line 597 of file `mutex`.

The documentation for this struct was generated from the following file:

- [mutex](#)

5.920 `std::ostream_iterator< _Tp, _CharT, _Traits >` Class Template Reference

Inheritance diagram for `std::ostream_iterator< _Tp, _CharT, _Traits >`:



Public Types

- typedef void [difference_type](#)
- typedef [output_iterator_tag](#) [iterator_category](#)
- typedef void [pointer](#)
- typedef void [reference](#)
- typedef void [value_type](#)

- typedef `_CharT` [char_type](#)
- typedef `_Traits` [traits_type](#)
- typedef `basic_ostream< _CharT, _Traits >` [ostream_type](#)

Public Member Functions

- [ostream_iterator](#) ([ostream_type](#) &__s)
- [ostream_iterator](#) ([ostream_type](#) &__s, const [_CharT](#) *__c)
- [ostream_iterator](#) (const [ostream_iterator](#) &__obj)
- [ostream_iterator](#) & **operator*** ()
- [ostream_iterator](#) & **operator++** ()
- [ostream_iterator](#) & **operator++** (int)
- [ostream_iterator](#) & **operator=** (const [_Tp](#) &__value)

5.920.1 Detailed Description

```
template<typename _Tp, typename _CharT = char, typename _Traits = char_traits<_CharT>>
class std::ostream_iterator< _Tp, _CharT, _Traits >
```

Provides output iterator semantics for streams.

This class provides an iterator to write to an ostream. The type `Tp` is the only type written by this iterator and there must be an `operator<<(Tp)` defined.

Template Parameters

<code>_Tp</code>	The type to write to the ostream.
<code>_CharT</code>	The ostream char_type.
<code>_Traits</code>	The ostream char_traits.

Definition at line 154 of file `stream_iterator.h`.

5.920.2 Member Typedef Documentation

5.920.2.1 `template<typename _Tp, typename _CharT = char, typename _Traits = char_traits<_CharT>> typedef _CharT std::ostream_iterator< _Tp, _CharT, _Traits >::char_type`

Public typedef.

Definition at line 160 of file `stream_iterator.h`.

5.920.2.2 `typedef void std::iterator< output_iterator_tag, void, void, void, void >::difference_type` `[inherited]`

Distance between iterators is represented as this type.

Definition at line 125 of file `stl_iterator_base_types.h`.

5.920.2.3 `typedef output_iterator_tag std::iterator< output_iterator_tag, void, void, void, void >::iterator_category` `[inherited]`

One of the [tag types](#).

Definition at line 121 of file `stl_iterator_base_types.h`.

```
5.920.2.4 template<typename _Tp, typename _CharT = char, typename _Traits = char_traits<_CharT>> typedef
    basic_ostream<_CharT, _Traits> std::ostream_iterator< _Tp, _CharT, _Traits >::ostream_type
```

Public typedef.

Definition at line 162 of file stream_iterator.h.

```
5.920.2.5 typedef void std::iterator< output_iterator_tag, void, void, void, void >::pointer [inherited]
```

This type represents a pointer-to-value_type.

Definition at line 127 of file stl_iterator_base_types.h.

```
5.920.2.6 typedef void std::iterator< output_iterator_tag, void, void, void, void >::reference [inherited]
```

This type represents a reference-to-value_type.

Definition at line 129 of file stl_iterator_base_types.h.

```
5.920.2.7 template<typename _Tp, typename _CharT = char, typename _Traits = char_traits<_CharT>> typedef _Traits
    std::ostream_iterator< _Tp, _CharT, _Traits >::traits_type
```

Public typedef.

Definition at line 161 of file stream_iterator.h.

```
5.920.2.8 typedef void std::iterator< output_iterator_tag, void, void, void, void >::value_type [inherited]
```

The type "pointed to" by the iterator.

Definition at line 123 of file stl_iterator_base_types.h.

5.920.3 Constructor & Destructor Documentation

```
5.920.3.1 template<typename _Tp, typename _CharT = char, typename _Traits = char_traits<_CharT>>
    std::ostream_iterator< _Tp, _CharT, _Traits >::ostream_iterator ( ostream_type &__s ) [inline]
```

Construct from an ostream.

Definition at line 171 of file stream_iterator.h.

```
5.920.3.2 template<typename _Tp, typename _CharT = char, typename _Traits = char_traits<_CharT>>
    std::ostream_iterator< _Tp, _CharT, _Traits >::ostream_iterator ( ostream_type &__s, const _CharT * __c )
    [inline]
```

Construct from an ostream.

The delimiter string *c* is written to the stream after every *Tp* written to the stream. The delimiter is not copied, and thus must not be destroyed while this iterator is in use.

Parameters

<code>_↔ _s</code>	Underlying ostream to write to.
<code>_↔ _c</code>	CharT delimiter string to insert.

Definition at line 183 of file stream_iterator.h.

```
5.920.3.3  template<typename _Tp , typename _CharT = char, typename _Traits = char_traits<_CharT>>
            std::ostream_iterator< _Tp, _CharT, _Traits >::ostream_iterator ( const ostream_iterator< _Tp, _CharT,
            _Traits > &__obj )  [inline]
```

Copy constructor.

Definition at line 187 of file stream_iterator.h.

5.920.4 Member Function Documentation

```
5.920.4.1  template<typename _Tp , typename _CharT = char, typename _Traits = char_traits<_CharT>> ostream_iterator&
            std::ostream_iterator< _Tp, _CharT, _Traits >::operator= ( const _Tp &__value )  [inline]
```

Writes *value* to underlying ostream using operator<<. If constructed with delimiter string, writes delimiter to ostream.

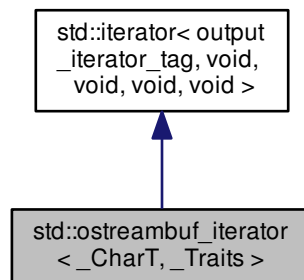
Definition at line 193 of file stream_iterator.h.

The documentation for this class was generated from the following file:

- [stream_iterator.h](#)

5.921 std::ostreambuf_iterator< _CharT, _Traits > Class Template Reference

Inheritance diagram for std::ostreambuf_iterator< _CharT, _Traits >:



Public Types

- typedef void [difference_type](#)
- typedef [output_iterator_tag](#) [iterator_category](#)
- typedef void [pointer](#)
- typedef void [reference](#)
- typedef void [value_type](#)

- typedef [_CharT](#) [char_type](#)
- typedef [_Traits](#) [traits_type](#)
- typedef [basic_streambuf](#)< [_CharT](#), [_Traits](#) > [streambuf_type](#)
- typedef [basic_ostream](#)< [_CharT](#), [_Traits](#) > [ostream_type](#)

Public Member Functions

- [ostreambuf_iterator](#) ([ostream_type](#) &__s) noexcept
- [ostreambuf_iterator](#) ([streambuf_type](#) *__s) noexcept
- [ostreambuf_iterator](#) & [M_put](#) (const [_CharT](#) *__ws, [streamsize](#) __len)
- bool [failed](#) () const noexcept
- [ostreambuf_iterator](#) & [operator*](#) ()
- [ostreambuf_iterator](#) & [operator++](#) (int)
- [ostreambuf_iterator](#) & [operator++](#) ()
- [ostreambuf_iterator](#) & [operator=](#) ([_CharT](#) __c)

Friends

- template<typename [_CharT2](#) >
[__gnu_cxx::__enable_if](#)< [__is_char](#)< [_CharT2](#) >::__value, [ostreambuf_iterator](#)< [_CharT2](#) >::__type **copy**
([istreambuf_iterator](#)< [_CharT2](#) >, [istreambuf_iterator](#)< [_CharT2](#) >, [ostreambuf_iterator](#)< [_CharT2](#) >)

5.921.1 Detailed Description

```
template<typename _CharT, typename _Traits>
class std::ostreambuf_iterator< _CharT, _Traits >
```

Provides output iterator semantics for streambufs.

Definition at line 128 of file iosfwd.

5.921.2 Member Typedef Documentation

5.921.2.1 template<typename [_CharT](#), typename [_Traits](#)> typedef [_CharT](#) [std::ostreambuf_iterator](#)< [_CharT](#), [_Traits](#) >::__char_type

Public typedefs.

Definition at line 223 of file streambuf_iterator.h.

5.921.2.2 **typedef void std::iterator< output_iterator_tag , void , void , void , void >::difference_type** [inherited]

Distance between iterators is represented as this type.

Definition at line 125 of file stl_iterator_base_types.h.

5.921.2.3 **typedef output_iterator_tag std::iterator< output_iterator_tag , void , void , void , void >::iterator_category** [inherited]

One of the [tag types](#).

Definition at line 121 of file stl_iterator_base_types.h.

5.921.2.4 **template<typename _CharT, typename _Traits> typedef basic_ostream<_CharT, _Traits> std::ostreambuf_iterator< _CharT, _Traits >::ostream_type**

Public typedefs.

Definition at line 226 of file streambuf_iterator.h.

5.921.2.5 **typedef void std::iterator< output_iterator_tag , void , void , void , void >::pointer** [inherited]

This type represents a pointer-to-value_type.

Definition at line 127 of file stl_iterator_base_types.h.

5.921.2.6 **typedef void std::iterator< output_iterator_tag , void , void , void , void >::reference** [inherited]

This type represents a reference-to-value_type.

Definition at line 129 of file stl_iterator_base_types.h.

5.921.2.7 **template<typename _CharT, typename _Traits> typedef basic_streambuf<_CharT, _Traits> std::ostreambuf_iterator< _CharT, _Traits >::streambuf_type**

Public typedefs.

Definition at line 225 of file streambuf_iterator.h.

5.921.2.8 **template<typename _CharT, typename _Traits> typedef _Traits std::ostreambuf_iterator< _CharT, _Traits >::traits_type**

Public typedefs.

Definition at line 224 of file streambuf_iterator.h.

5.921.2.9 **typedef void std::iterator< output_iterator_tag , void , void , void , void >::value_type** [inherited]

The type "pointed to" by the iterator.

Definition at line 123 of file stl_iterator_base_types.h.

5.921.3 Constructor & Destructor Documentation

5.921.3.1 `template<typename _CharT, typename _Traits> std::ostreambuf_iterator< _CharT, _Traits >::ostreambuf_iterator (ostream_type &__s) [inline], [noexcept]`

Construct output iterator from ostream.

Definition at line 241 of file streambuf_iterator.h.

5.921.3.2 `template<typename _CharT, typename _Traits> std::ostreambuf_iterator< _CharT, _Traits >::ostreambuf_iterator (streambuf_type *__s) [inline], [noexcept]`

Construct output iterator from streambuf.

Definition at line 245 of file streambuf_iterator.h.

5.921.4 Member Function Documentation

5.921.4.1 `template<typename _CharT, typename _Traits> bool std::ostreambuf_iterator< _CharT, _Traits >::failed () const [inline], [noexcept]`

Return true if previous operator=() failed.

Definition at line 275 of file streambuf_iterator.h.

References std::basic_streambuf< _CharT, _Traits >::egptr(), std::basic_streambuf< _CharT, _Traits >::gptr(), std::basic_streambuf< _CharT, _Traits >::sgetc(), std::basic_streambuf< _CharT, _Traits >::snextc(), std::basic_streambuf< _CharT, _Traits >::sputn(), and std::basic_streambuf< _CharT, _Traits >::underflow().

5.921.4.2 `template<typename _CharT, typename _Traits> ostreambuf_iterator& std::ostreambuf_iterator< _CharT, _Traits >::operator* () [inline]`

Return *this.

Definition at line 260 of file streambuf_iterator.h.

5.921.4.3 `template<typename _CharT, typename _Traits> ostreambuf_iterator& std::ostreambuf_iterator< _CharT, _Traits >::operator++ (int) [inline]`

Return *this.

Definition at line 265 of file streambuf_iterator.h.

5.921.4.4 `template<typename _CharT, typename _Traits> ostreambuf_iterator& std::ostreambuf_iterator< _CharT, _Traits >::operator++ () [inline]`

Return *this.

Definition at line 270 of file streambuf_iterator.h.

5.921.4.5 `template<typename _CharT, typename _Traits> ostreambuf_iterator& std::ostreambuf_iterator< _CharT, _Traits >::operator= (_CharT __c) [inline]`

Write character to streambuf. Calls streambuf.sputc().

Definition at line 250 of file streambuf_iterator.h.

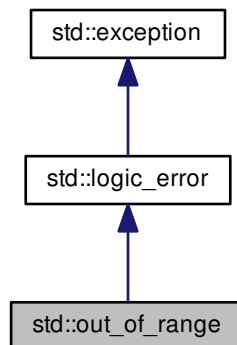
References `std::basic_streambuf< _CharT, _Traits >::sputc()`.

The documentation for this class was generated from the following files:

- [iosfwd](#)
- [streambuf_iterator.h](#)

5.922 std::out_of_range Class Reference

Inheritance diagram for `std::out_of_range`:



Public Member Functions

- **out_of_range** (const [string](#) &__arg) _GLIBCXX_TXN_SAFE
- **out_of_range** (const char *) _GLIBCXX_TXN_SAFE
- virtual const char * [what](#) () const _GLIBCXX_TXN_SAFE_DYN noexcept

5.922.1 Detailed Description

This represents an argument whose value is not within the expected range (e.g., boundary checks in `basic_string`).

Definition at line 182 of file `stdexcept`.

5.922.2 Member Function Documentation

5.922.2.1 `virtual const char* std::logic_error::what () const` `[virtual]`, `[noexcept]`, `[inherited]`

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

Reimplemented in [std::future_error](#).

The documentation for this class was generated from the following file:

- [stdexcept](#)

5.923 `std::output_iterator_tag` Struct Reference

5.923.1 Detailed Description

Marking output iterators.

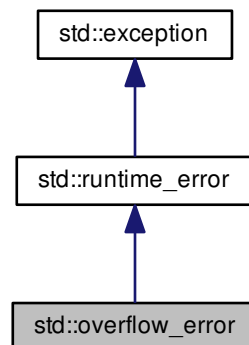
Definition at line 92 of file `stl_iterator_base_types.h`.

The documentation for this struct was generated from the following file:

- [stl_iterator_base_types.h](#)

5.924 `std::overflow_error` Class Reference

Inheritance diagram for `std::overflow_error`:



Public Member Functions

- **overflow_error** (const [string](#) &__arg) _GLIBCXX_TXN_SAFE
- **overflow_error** (const char *) _GLIBCXX_TXN_SAFE
- virtual const char * [what](#) () const _GLIBCXX_TXN_SAFE_DYN noexcept

5.924.1 Detailed Description

Thrown to indicate arithmetic overflow.

Definition at line 241 of file `stdexcept`.

5.924.2 Member Function Documentation

5.924.2.1 `virtual const char* std::runtime_error::what () const` `[virtual],[noexcept],[inherited]`

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

The documentation for this class was generated from the following file:

- [stdexcept](#)

5.925 `std::owner_less<_Tp>` Struct Template Reference

5.925.1 Detailed Description

```
template<typename _Tp>
struct std::owner_less<_Tp>
```

Primary template `owner_less`.

Definition at line 539 of file `bits/shared_ptr.h`.

The documentation for this struct was generated from the following file:

- [bits/shared_ptr.h](#)

5.926 `std::owner_less< shared_ptr<_Tp> >` Struct Template Reference

Inherits `std::_Sp_owner_less<_Tp, _Tp1>`.

Public Types

- `typedef _Tp first_argument_type`
- `typedef bool result_type`
- `typedef _Tp second_argument_type`

Public Member Functions

- `bool operator() (const _Tp &__lhs, const _Tp &__rhs) const`
- `bool operator() (const _Tp &__lhs, const _Tp1 &__rhs) const`
- `bool operator() (const _Tp1 &__lhs, const _Tp &__rhs) const`

5.926.1 Detailed Description

```
template<typename _Tp>
struct std::owner_less< shared_ptr< _Tp > >
```

Partial specialization of `owner_less` for `shared_ptr`.

Definition at line 543 of file `bits/shared_ptr.h`.

5.926.2 Member Typedef Documentation

5.926.2.1 `typedef _Tp std::binary_function< _Tp, _Tp, bool >::first_argument_type` [inherited]

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.926.2.2 `typedef bool std::binary_function< _Tp, _Tp, bool >::result_type` [inherited]

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.926.2.3 `typedef _Tp std::binary_function< _Tp, _Tp, bool >::second_argument_type` [inherited]

`second_argument_type` is the type of the second argument

Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [bits/shared_ptr.h](#)

5.927 `std::owner_less< weak_ptr< _Tp > >` Struct Template Reference

Inherits `std::_Sp_owner_less< _Tp, _Tp1 >`.

Public Types

- typedef `_Tp` [first_argument_type](#)
- typedef `bool` [result_type](#)
- typedef `_Tp` [second_argument_type](#)

Public Member Functions

- `bool` **operator()** (`const _Tp &__lhs, const _Tp &__rhs`) `const`
- `bool` **operator()** (`const _Tp &__lhs, const _Tp1 &__rhs`) `const`
- `bool` **operator()** (`const _Tp1 &__lhs, const _Tp &__rhs`) `const`

5.927.1 Detailed Description

```
template<typename _Tp>
struct std::owner_less< weak_ptr< _Tp > >
```

Partial specialization of `owner_less` for `weak_ptr`.

Definition at line 549 of file `bits/shared_ptr.h`.

5.927.2 Member Typedef Documentation

5.927.2.1 `typedef _Tp std::binary_function< _Tp, _Tp, bool >::first_argument_type` `[inherited]`

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.927.2.2 `typedef bool std::binary_function< _Tp, _Tp, bool >::result_type` `[inherited]`

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.927.2.3 `typedef _Tp std::binary_function< _Tp, _Tp, bool >::second_argument_type` `[inherited]`

`second_argument_type` is the type of the second argument

Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [bits/shared_ptr.h](#)

5.928 `std::packaged_task<_Res(_ArgTypes...)>` Class Template Reference

Public Member Functions

- `template<typename _Allocator >`
`packaged_task` (`allocator_arg_t`, `const _Allocator &__a`) `noexcept`
- `template<typename _Fn, typename = typename __constrain_pkgdtask<packaged_task, _Fn>::__type>`
`packaged_task` (`_Fn &&__fn`)
- `template<typename _Fn, typename _Alloc, typename = typename __constrain_pkgdtask<packaged_task, _Fn>::__type>`
`packaged_task` (`allocator_arg_t`, `const _Alloc &__a`, `_Fn &&__fn`)
- `packaged_task` (`const packaged_task &`)=`delete`
- `template<typename _Allocator >`
`packaged_task` (`allocator_arg_t`, `const _Allocator &`, `const packaged_task &`)=`delete`
- `packaged_task` (`packaged_task &&__other`) `noexcept`
- `template<typename _Allocator >`
`packaged_task` (`allocator_arg_t`, `const _Allocator &`, `packaged_task &&__other`) `noexcept`
- `future<_Res >` `get_future` ()
- `void make_ready_at_thread_exit` (`_ArgTypes...__args`)
- `void operator()` (`_ArgTypes...__args`)
- `packaged_task & operator=` (`const packaged_task &`)=`delete`
- `packaged_task & operator=` (`packaged_task &&__other`) `noexcept`
- `void reset` ()
- `void swap` (`packaged_task &__other`) `noexcept`
- `bool valid` () `const noexcept`

5.928.1 Detailed Description

```
template<typename _Res, typename... _ArgTypes>
class std::packaged_task<_Res(_ArgTypes...)>
```

`packaged_task`

Definition at line 1473 of file `future`.

The documentation for this class was generated from the following file:

- `future`

5.929 `std::pair<_T1,_T2 >` Struct Template Reference

Public Types

- `template<typename _U1, typename _U2 >`
using `_PCCFP` = `_PCC<!is_same<_T1, _U1 >::value||!is_same<_T2, _U2 >::value, _T1, _T2 >`
- using `_PCCP` = `_PCC< true, _T1, _T2 >`
- `typedef _T1 first_type`
- `typedef _T2 second_type`

Public Member Functions

- `template<typename _U1 = _T1, typename _U2 = _T2, typename enable_if< __and< __is_implicitly_default_constructible< _U1 >, __is_implicitly_default_constructible< _U2 >> ::value, bool >::type = true>`
`constexpr pair ()`
- `template<typename _U1 = _T1, typename _U2 = _T2, typename enable_if< _PCCP::template _ConstructiblePair< _U1, _U2 >() && _PCCP::template _ImplicitlyConvertiblePair< _U1, _U2 >(), bool >::type = true>`
`constexpr pair (const _T1 &__a, const _T2 &__b)`
- `template<typename _U1 = _T1, typename _U2 = _T2, typename enable_if< _PCCP::template _ConstructiblePair< _U1, _U2 >() && !_PCCP::template _ImplicitlyConvertiblePair< _U1, _U2 >(), bool >::type = false>`
`constexpr pair (const _T1 &__a, const _T2 &__b)`
- `template<typename _U1, typename _U2, typename enable_if< _PCCFP< _U1, _U2 >::template _ConstructiblePair< _U1, _U2 >() && _PCCFP< _U1, _U2 >::template _ImplicitlyConvertiblePair< _U1, _U2 >(), bool >::type = true>`
`constexpr pair (const pair< _U1, _U2 > &__p)`
- `template<typename _U1, typename _U2, typename enable_if< _PCCFP< _U1, _U2 >::template _ConstructiblePair< _U1, _U2 >() && !_PCCFP< _U1, _U2 >::template _ImplicitlyConvertiblePair< _U1, _U2 >(), bool >::type = false>`
`constexpr pair (const pair< _U1, _U2 > &__p)`
- `constexpr pair (const pair &)=default`
- `constexpr pair (pair &&)=default`
- `template<typename _U1, typename enable_if< _PCCP::template _MoveCopyPair< true, _U1, _T2 >(), bool >::type = true>`
`constexpr pair (_U1 &&__x, const _T2 &__y)`
- `template<typename _U1, typename enable_if< _PCCP::template _MoveCopyPair< false, _U1, _T2 >(), bool >::type = false>`
`constexpr pair (_U1 &&__x, const _T2 &__y)`
- `template<typename _U2, typename enable_if< _PCCP::template _CopyMovePair< true, _T1, _U2 >(), bool >::type = true>`
`constexpr pair (const _T1 &__x, _U2 &&__y)`
- `template<typename _U2, typename enable_if< _PCCP::template _CopyMovePair< false, _T1, _U2 >(), bool >::type = false>`
`pair (const _T1 &__x, _U2 &&__y)`
- `template<typename _U1, typename _U2, typename enable_if< _PCCP::template _MoveConstructiblePair< _U1, _U2 >() && _PCCP::template _ImplicitlyMoveConvertiblePair< _U1, _U2 >(), bool >::type = true>`
`constexpr pair (_U1 &&__x, _U2 &&__y)`
- `template<typename _U1, typename _U2, typename enable_if< _PCCP::template _MoveConstructiblePair< _U1, _U2 >() && !_PCCP::template _ImplicitlyMoveConvertiblePair< _U1, _U2 >(), bool >::type = false>`
`constexpr pair (_U1 &&__x, _U2 &&__y)`
- `template<typename _U1, typename _U2, typename enable_if< _PCCFP< _U1, _U2 >::template _MoveConstructiblePair< _U1, _U2 >() && _PCCFP< _U1, _U2 >::template _ImplicitlyMoveConvertiblePair< _U1, _U2 >(), bool >::type = true>`
`constexpr pair (pair< _U1, _U2 > &&__p)`
- `template<typename _U1, typename _U2, typename enable_if< _PCCFP< _U1, _U2 >::template _MoveConstructiblePair< _U1, _U2 >() && !_PCCFP< _U1, _U2 >::template _ImplicitlyMoveConvertiblePair< _U1, _U2 >(), bool >::type = false>`
`constexpr pair (pair< _U1, _U2 > &&__p)`
- `template<typename... _Args1, typename... _Args2>`
`pair (piecewise_construct_t, tuple< _Args1... >, tuple< _Args2... >)`
- `pair & operator= (typename conditional< __and< is_copy_assignable< _T1 >, is_copy_assignable< _T2 >>::value, const pair &, const __nonesuch & >::type __p)`
- `pair & operator= (typename conditional< __not< __and< is_copy_assignable< _T1 >, is_copy_assignable< _T2 >>::value, const pair &, const __nonesuch & >::type __p)=delete`
- `pair & operator= (typename conditional< __and< is_move_assignable< _T1 >, is_move_assignable< _T2 >>::value, pair &&, __nonesuch && >::type __p) noexcept(__and< is_nothrow_move_assignable< _T1 >, is_nothrow_move_assignable< _T2 >>::value)`
- `template<typename _U1, typename _U2 >`
`enable_if< __and< is_assignable< _T1 &, const _U1 & >, is_assignable< _T2 &, const _U2 & >>::value,`
`pair & >::type operator= (const pair< _U1, _U2 > &__p)`
- `template<typename _U1, typename _U2 >`
`enable_if< __and< is_assignable< _T1 &, _U1 && >, is_assignable< _T2 &, _U2 && >>::value, pair &`
`>::type operator= (pair< _U1, _U2 > &&__p)`

- void **swap** (`pair` &__p) noexcept(__is_nothrow_swappable<_T1>::value &&__is_nothrow_swappable<_T2>::value)

Public Attributes

- `_T1` `first`
- `_T2` `second`

5.929.1 Detailed Description

```
template<typename _T1, typename _T2>
struct std::pair<_T1, _T2>
```

Struct holding two objects of arbitrary type.

Template Parameters

<code>_T1</code>	Type of first object.
<code>_T2</code>	Type of second object.

Definition at line 190 of file `stl_pair.h`.

5.929.2 Member Typedef Documentation

5.929.2.1 `template<typename _T1, typename _T2> template<typename _U1, typename _U2> using std::pair<_T1, _T2>::_PCCFP = _PCC<!is_same<_T1, _U1>::value || !is_same<_T2, _U2>::value, _T1, _T2>`

There is also a templated copy ctor for the `pair` class itself.

Definition at line 264 of file `stl_pair.h`.

5.929.2.2 `template<typename _T1, typename _T2> using std::pair<_T1, _T2>::_PCCP = _PCC<true, _T1, _T2>`

Two objects may be passed to a `pair` constructor to be copied.

Definition at line 233 of file `stl_pair.h`.

5.929.2.3 `template<typename _T1, typename _T2> typedef _T2 std::pair<_T1, _T2>::second_type`

`first_type` is the first bound type

Definition at line 193 of file `stl_pair.h`.

5.929.3 Constructor & Destructor Documentation

5.929.3.1 `template<typename _T1, typename _T2> template<typename _U1 = _T1, typename _U2 = _T2, typename enable_if<__and<__is_implicitly_default_constructible<_U1>, __is_implicitly_default_constructible<_U2>>::value, bool>::type = true> constexpr std::pair<_T1, _T2>::pair () [inline]`

`second` is a copy of the second object

The default constructor creates `first` and `second` using their respective default constructors.

Definition at line 210 of file `stl_pair.h`.

5.929.4 Member Data Documentation

5.929.4.1 `template<typename _T1, typename _T2> _T1 std::pair<_T1, _T2>::first`

`second_type` is the second bound type

Definition at line 195 of file `stl_pair.h`.

Referenced by `__gnu_parallel::__qsb_conquer()`, `__gnu_debug::__valid_range_aux()`, `std::_Temporary_buffer<_ForwardIterator, _Tp>::_Temporary_buffer()`, `std::set<_Key, _Compare, _Alloc>::insert()`, `std::minmax_element()`, `__gnu_pbds::detail::pat_trie_map<Key, Mapped, Node_And_It_Traits, _Alloc>::node_end()`, `std::operator<()`, `std::operator==()`, and `std::regex_replace()`.

5.929.4.2 `template<typename _T1, typename _T2> _T2 std::pair<_T1, _T2>::second`

`first` is a copy of the first object

Definition at line 196 of file `stl_pair.h`.

Referenced by `__gnu_debug::__get_distance()`, `__gnu_parallel::__qsb_conquer()`, `__gnu_debug::__valid_range_aux()`, `std::_Temporary_buffer<_ForwardIterator, _Tp>::_Temporary_buffer()`, `std::set<_Key, _Compare, _Alloc>::insert()`, `std::minmax_element()`, `std::operator==()`, and `std::regex_replace()`.

The documentation for this struct was generated from the following files:

- [stl_pair.h](#)
- [tuple](#)

5.930 `std::piecewise_constant_distribution<_RealType>` Class Template Reference

Classes

- struct [param_type](#)

Public Types

- typedef _RealType [result_type](#)

Public Member Functions

- template<typename _InputIteratorB, typename _InputIteratorW >
piecewise_constant_distribution (_InputIteratorB __bfirst, _InputIteratorB __bend, _InputIteratorW __wbegin)
- template<typename _Func >
piecewise_constant_distribution ([initializer_list](#)< _RealType > __bl, _Func __fw)
- template<typename _Func >
piecewise_constant_distribution (size_t __nw, _RealType __xmin, _RealType __xmax, _Func __fw)
- **piecewise_constant_distribution** (const [param_type](#) &__p)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- template<typename _UniformRandomNumberGenerator >
void **generate** ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [std::vector](#)< double > **densities** () const
- [std::vector](#)< _RealType > **intervals** () const
- [result_type](#) **max** () const
- [result_type](#) **min** () const
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [param_type](#) **param** () const
- void **param** (const [param_type](#) &__param)
- void **reset** ()

Friends

- template<typename _RealType1, typename _CharT, typename _Traits >
[std::basic_ostream](#)< _CharT, _Traits > & **operator<<** ([std::basic_ostream](#)< _CharT, _Traits > &__os, const [std::piecewise_constant_distribution](#)< _RealType1 > &__x)
- bool **operator==** (const [piecewise_constant_distribution](#) &__d1, const [piecewise_constant_distribution](#) &__d2)
- template<typename _RealType1, typename _CharT, typename _Traits >
[std::basic_istream](#)< _CharT, _Traits > & **operator>>** ([std::basic_istream](#)< _CharT, _Traits > &__is, [std::piecewise_constant_distribution](#)< _RealType1 > &__x)

5.930.1 Detailed Description

```
template<typename _RealType = double>
class std::piecewise_constant_distribution< _RealType >
```

A `piecewise_constant_distribution` random number distribution.

The formula for the piecewise constant probability mass function is

Definition at line 5316 of file `random.h`.

5.930.2 Member Typedef Documentation

5.930.2.1 `template<typename _RealType = double> typedef _RealType std::piecewise_constant_distribution< _RealType >::result_type`

The type of the range of the distribution.

Definition at line 5319 of file random.h.

5.930.3 Member Function Documentation

5.930.3.1 `template<typename _RealType = double> std::vector<double> std::piecewise_constant_distribution< _RealType >::densities () const [inline]`

Returns a vector of the probability densities.

Definition at line 5437 of file random.h.

Referenced by `std::operator>>()`.

5.930.3.2 `template<typename _RealType = double> std::vector<_RealType> std::piecewise_constant_distribution< _RealType >::intervals () const [inline]`

Returns a vector of the intervals.

Definition at line 5421 of file random.h.

Referenced by `std::operator>>()`.

5.930.3.3 `template<typename _RealType = double> result_type std::piecewise_constant_distribution< _RealType >::max () const [inline]`

Returns the least upper bound value of the distribution.

Definition at line 5472 of file random.h.

5.930.3.4 `template<typename _RealType = double> result_type std::piecewise_constant_distribution< _RealType >::min () const [inline]`

Returns the greatest lower bound value of the distribution.

Definition at line 5462 of file random.h.

5.930.3.5 `template<typename _RealType = double> template<typename UniformRandomNumberGenerator > result_type std::piecewise_constant_distribution< _RealType >::operator() (_UniformRandomNumberGenerator & __urng) [inline]`

Generating functions.

Definition at line 5483 of file random.h.

Referenced by `std::operator>>()`.

5.930.3.6 `template<typename _RealType = double> param_type std::piecewise_constant_distribution<_RealType>::param () const` `[inline]`

Returns the parameter set of the distribution.

Definition at line 5447 of file random.h.

Referenced by std::operator>>().

5.930.3.7 `template<typename _RealType = double> void std::piecewise_constant_distribution<_RealType>::param (const param_type & __param)` `[inline]`

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

Definition at line 5455 of file random.h.

5.930.3.8 `template<typename _RealType = double> void std::piecewise_constant_distribution<_RealType>::reset ()` `[inline]`

Resets the distribution state.

Definition at line 5414 of file random.h.

5.930.4 Friends And Related Function Documentation

5.930.4.1 `template<typename _RealType = double> template<typename _RealType1, typename _CharT, typename _Traits> std::basic_ostream<_CharT, _Traits>& operator<< (std::basic_ostream<_CharT, _Traits> & __os, const std::piecewise_constant_distribution<_RealType1> & __x)` `[friend]`

Inserts a piecewise_constant_distribution random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A piecewise_constant_distribution random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

5.930.4.2 `template<typename _RealType = double> bool operator==(const piecewise_constant_distribution<_RealType> & __d1, const piecewise_constant_distribution<_RealType> & __d2)` `[friend]`

Return true if two piecewise constant distributions have the same parameters.

Definition at line 5518 of file random.h.

5.930.4.3 `template<typename _RealType = double> template<typename _RealType1, typename _CharT, typename _Traits
> std::basic_istream<_CharT, _Traits>& operator>> (std::basic_istream<_CharT, _Traits> & __is,
std::piecewise_constant_distribution<_RealType1> & __x) [friend]`

Extracts a `piecewise_constant_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>piecewise_constant_distribution</code> random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.931 `std::piecewise_constant_distribution<_RealType>::param_type` Struct Reference

Public Types

- typedef `piecewise_constant_distribution<_RealType>` **distribution_type**

Public Member Functions

- `template<typename _InputIteratorB, typename _InputIteratorW>
param_type (_InputIteratorB __bfirst, _InputIteratorB __bend, _InputIteratorW __wbegin)`
- `template<typename _Func>
param_type (initializer_list<_RealType> __bi, _Func __fw)`
- `template<typename _Func>
param_type (size_t __nw, _RealType __xmin, _RealType __xmax, _Func __fw)`
- `param_type (const param_type &)=default`
- `std::vector<double> densities () const`
- `std::vector<_RealType> intervals () const`
- `param_type & operator= (const param_type &)=default`

Friends

- `bool operator== (const param_type &__p1, const param_type &__p2)`
- `class piecewise_constant_distribution<_RealType>`

5.931.1 Detailed Description

```
template<typename _RealType = double>
struct std::piecewise_constant_distribution< _RealType >::param_type
```

Parameter type.

Definition at line 5325 of file `random.h`.

The documentation for this struct was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.932 `std::piecewise_construct_t` Struct Reference

5.932.1 Detailed Description

`piecewise_construct_t`

Definition at line 76 of file `stl_pair.h`.

The documentation for this struct was generated from the following file:

- [stl_pair.h](#)

5.933 `std::piecewise_linear_distribution< _RealType >` Class Template Reference

Classes

- struct [param_type](#)

Public Types

- typedef `_RealType` [result_type](#)

Public Member Functions

- `template<typename _InputIteratorB, typename _InputIteratorW >`
piecewise_linear_distribution (`_InputIteratorB __bfirst, _InputIteratorB __bend, _InputIteratorW __wbegin`)
- `template<typename _Func >`
piecewise_linear_distribution (`initializer_list<_RealType> __bl, _Func __fw`)
- `template<typename _Func >`
piecewise_linear_distribution (`size_t __nw, _RealType __xmin, _RealType __xmax, _Func __fw`)
- **piecewise_linear_distribution** (`const param_type &__p`)
- `template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >`
void __generate (`_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng`)
- `template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >`
void __generate (`_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const param_type &__p`)
- `template<typename _UniformRandomNumberGenerator >`
void __generate (`result_type *__f, result_type *__t, _UniformRandomNumberGenerator &__urng, const param_type &__p`)
- `std::vector<double> densities` () const
- `std::vector<_RealType> intervals` () const
- `result_type max` () const
- `result_type min` () const
- `template<typename _UniformRandomNumberGenerator >`
result_type operator() (`_UniformRandomNumberGenerator &__urng`)
- `template<typename _UniformRandomNumberGenerator >`
result_type operator() (`_UniformRandomNumberGenerator &__urng, const param_type &__p`)
- `param_type param` () const
- `void param` (`const param_type &__param`)
- `void reset` ()

Friends

- `template<typename _RealType1, typename _CharT, typename _Traits >`
std::basic_ostream< `_CharT, _Traits` > & **operator<<** (`std::basic_ostream`< `_CharT, _Traits` > &__os, const `std::piecewise_linear_distribution`< `_RealType1` > &__x)
- `bool operator==` (`const piecewise_linear_distribution &__d1, const piecewise_linear_distribution &__d2`)
- `template<typename _RealType1, typename _CharT, typename _Traits >`
std::basic_istream< `_CharT, _Traits` > & **operator>>** (`std::basic_istream`< `_CharT, _Traits` > &__is, `std::piecewise_linear_distribution`< `_RealType1` > &__x)

5.933.1 Detailed Description

```
template<typename _RealType = double>
class std::piecewise_linear_distribution< _RealType >
```

A `piecewise_linear_distribution` random number distribution.

The formula for the piecewise linear probability mass function is

Definition at line 5583 of file `random.h`.

5.933.2 Member Typedef Documentation

5.933.2.1 `template<typename _RealType = double> typedef _RealType std::piecewise_linear_distribution<_RealType>::result_type`

The type of the range of the distribution.

Definition at line 5586 of file random.h.

5.933.3 Member Function Documentation

5.933.3.1 `template<typename _RealType = double> std::vector<double> std::piecewise_linear_distribution<_RealType>::densities () const [inline]`

Return a vector of the probability densities of the distribution.

Definition at line 5707 of file random.h.

References `std::vector<_Tp, _Alloc>::empty()`.

5.933.3.2 `template<typename _RealType = double> std::vector<_RealType> std::piecewise_linear_distribution<_RealType>::intervals () const [inline]`

Return the intervals of the distribution.

Definition at line 5690 of file random.h.

References `std::vector<_Tp, _Alloc>::empty()`.

5.933.3.3 `template<typename _RealType = double> result_type std::piecewise_linear_distribution<_RealType>::max () const [inline]`

Returns the least upper bound value of the distribution.

Definition at line 5742 of file random.h.

References `std::vector<_Tp, _Alloc>::back()`, and `std::vector<_Tp, _Alloc>::empty()`.

5.933.3.4 `template<typename _RealType = double> result_type std::piecewise_linear_distribution<_RealType>::min () const [inline]`

Returns the greatest lower bound value of the distribution.

Definition at line 5732 of file random.h.

References `std::vector<_Tp, _Alloc>::empty()`, and `std::vector<_Tp, _Alloc>::front()`.

5.933.3.5 `template<typename _RealType = double> template<typename _UniformRandomNumberGenerator > result_type
std::piecewise_linear_distribution< _RealType >::operator() (_UniformRandomNumberGenerator & __urng)
[inline]`

Generating functions.

Definition at line 5753 of file random.h.

Referenced by `std::operator>>()`.

5.933.3.6 `template<typename _RealType = double> param_type std::piecewise_linear_distribution< _RealType
>::param () const [inline]`

Returns the parameter set of the distribution.

Definition at line 5717 of file random.h.

Referenced by `std::operator>>()`.

5.933.3.7 `template<typename _RealType = double> void std::piecewise_linear_distribution< _RealType >::param (const
param_type & __param) [inline]`

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

Definition at line 5725 of file random.h.

5.933.3.8 `template<typename _RealType = double> void std::piecewise_linear_distribution< _RealType >::reset ()
[inline]`

Resets the distribution state.

Definition at line 5683 of file random.h.

5.933.4 Friends And Related Function Documentation

5.933.4.1 `template<typename _RealType = double> template<typename _RealType1 , typename _CharT , typename _Traits >
std::basic_ostream< _CharT , _Traits > & operator<< (std::basic_ostream< _CharT , _Traits > & __os, const
std::piecewise_linear_distribution< _RealType1 > & __x) [friend]`

Inserts a `piecewise_linear_distribution` random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A <code>piecewise_linear_distribution</code> random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

5.933.4.2 `template<typename _RealType = double> bool operator==(const piecewise_linear_distribution<_RealType> & __d1, const piecewise_linear_distribution<_RealType> & __d2) [friend]`

Return true if two piecewise linear distributions have the same parameters.

Definition at line 5788 of file random.h.

5.933.4.3 `template<typename _RealType = double> template<typename _RealType1, typename _CharT, typename _Traits> > std::basic_istream<_CharT, _Traits>& operator>> (std::basic_istream<_CharT, _Traits> & __is, std::piecewise_linear_distribution<_RealType1> & __x) [friend]`

Extracts a `piecewise_linear_distribution` random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>piecewise_linear_distribution</code> random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.934 std::piecewise_linear_distribution<_RealType>::param_type Struct Reference

Public Types

- typedef [piecewise_linear_distribution<_RealType>](#) **distribution_type**

Public Member Functions

- `template<typename _InputIteratorB , typename _InputIteratorW >`
param_type (`_InputIteratorB __bfirst, _InputIteratorB __bend, _InputIteratorW __wbegin`)
- `template<typename _Func >`
param_type (`initializer_list<_RealType > __bl, _Func __fw`)
- `template<typename _Func >`
param_type (`size_t __nw, _RealType __xmin, _RealType __xmax, _Func __fw`)
- **param_type** (`const param_type &)=default`)
- `std::vector< double > densities () const`
- `std::vector< _RealType > intervals () const`
- `param_type & operator= (const param_type &)=default`

Friends

- `bool operator== (const param_type &__p1, const param_type &__p2)`
- `class piecewise_linear_distribution< _RealType >`

5.934.1 Detailed Description

```
template<typename _RealType = double>
struct std::piecewise_linear_distribution< _RealType >::param_type
```

Parameter type.

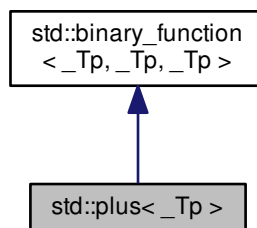
Definition at line 5592 of file random.h.

The documentation for this struct was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.935 std::plus<_Tp> Struct Template Reference

Inheritance diagram for `std::plus<_Tp>`:



Public Types

- typedef `_Tp` [first_argument_type](#)
- typedef `_Tp` [result_type](#)
- typedef `_Tp` [second_argument_type](#)

Public Member Functions

- `_GLIBCXX14_CONSTEXPR _Tp operator()` (`const _Tp &__x`, `const _Tp &__y`) `const`

5.935.1 Detailed Description

```
template<typename _Tp>
struct std::plus<_Tp>
```

One of the [math functors](#).

Definition at line 147 of file `stl_function.h`.

5.935.2 Member Typedef Documentation

5.935.2.1 `typedef _Tp std::binary_function<_Tp, _Tp, _Tp>::first_argument_type` `[inherited]`

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.935.2.2 `typedef _Tp std::binary_function<_Tp, _Tp, _Tp>::result_type` `[inherited]`

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.935.2.3 `typedef _Tp std::binary_function<_Tp, _Tp, _Tp>::second_argument_type` `[inherited]`

`second_argument_type` is the type of the second argument

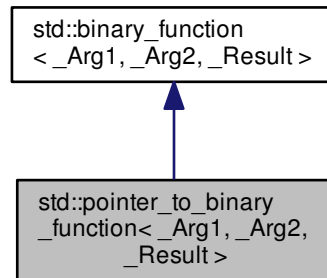
Definition at line 124 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.936 `std::pointer_to_binary_function< _Arg1, _Arg2, _Result >` Class Template Reference

Inheritance diagram for `std::pointer_to_binary_function< _Arg1, _Arg2, _Result >`:



Public Types

- typedef `_Arg1` [first_argument_type](#)
- typedef `_Result` [result_type](#)
- typedef `_Arg2` [second_argument_type](#)

Public Member Functions

- **`pointer_to_binary_function`** (`_Result(*__x)(_Arg1, _Arg2)`)
- `_Result` **`operator()`** (`_Arg1 __x, _Arg2 __y`) const

Protected Attributes

- `_Result(* _M_ptr)(_Arg1, _Arg2)`

5.936.1 Detailed Description

```
template<typename _Arg1, typename _Arg2, typename _Result>
class std::pointer_to_binary_function< _Arg1, _Arg2, _Result >
```

One of the [adaptors for function pointers](#).

Definition at line 843 of file `stl_function.h`.

5.936.2 Member Typedef Documentation

5.936.2.1 `template<typename _Arg1, typename _Arg2, typename _Result> typedef _Arg1 std::binary_function< _Arg1, _Arg2, _Result >::first_argument_type` [inherited]

`first_argument_type` is the type of the first argument

Definition at line 121 of file `stl_function.h`.

5.936.2.2 `template<typename _Arg1, typename _Arg2, typename _Result> typedef _Result std::binary_function< _Arg1, _Arg2, _Result >::result_type` [inherited]

`result_type` is the return type

Definition at line 127 of file `stl_function.h`.

5.936.2.3 `template<typename _Arg1, typename _Arg2, typename _Result> typedef _Arg2 std::binary_function< _Arg1, _Arg2, _Result >::second_argument_type` [inherited]

`second_argument_type` is the type of the second argument

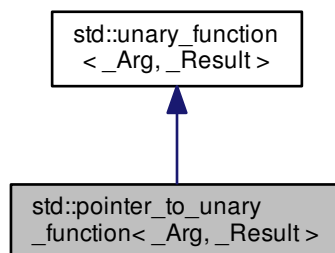
Definition at line 124 of file `stl_function.h`.

The documentation for this class was generated from the following file:

- [stl_function.h](#)

5.937 std::pointer_to_unary_function< _Arg, _Result > Class Template Reference

Inheritance diagram for `std::pointer_to_unary_function< _Arg, _Result >`:



Public Types

- typedef `_Arg` [argument_type](#)
- typedef `_Result` [result_type](#)

Public Member Functions

- **pointer_to_unary_function** (`_Result(*__x)(_Arg)`)
- `_Result` **operator()** (`_Arg __x`) const

Protected Attributes

- `_Result(* M_ptr)(_Arg)`

5.937.1 Detailed Description

```
template<typename _Arg, typename _Result>
class std::pointer_to_unary_function< _Arg, _Result >
```

One of the [adaptors for function pointers](#).

Definition at line 818 of file `stl_function.h`.

5.937.2 Member Typedef Documentation

5.937.2.1 `template<typename _Arg, typename _Result> typedef _Arg std::unary_function< _Arg, _Result >::argument_type` [\[inherited\]](#)

`argument_type` is the type of the argument

Definition at line 108 of file `stl_function.h`.

5.937.2.2 `template<typename _Arg, typename _Result> typedef _Result std::unary_function< _Arg, _Result >::result_type` [\[inherited\]](#)

`result_type` is the return type

Definition at line 111 of file `stl_function.h`.

The documentation for this class was generated from the following file:

- [stl_function.h](#)

5.938 std::pointer_traits<_Ptr> Struct Template Reference

Public Types

- using [difference_type](#) = __detected_or_t< ptrdiff_t, __difference_type, _Ptr >
- using [element_type](#) = __detected_or_t< __get_first_arg_t, __element_type, _Ptr >
- using [pointer](#) = _Ptr
- template<typename _Up >
using [rebind](#) = __detected_or_t< __replace_first_arg_t, __rebind, _Ptr, _Up >

Static Public Member Functions

- static _Ptr [pointer_to](#) (__make_not_void< [element_type](#) > &__e)

5.938.1 Detailed Description

```
template<typename _Ptr>
struct std::pointer_traits< _Ptr >
```

Uniform interface to all pointer-like types.

Definition at line 78 of file ptr_traits.h.

5.938.2 Member Typedef Documentation

5.938.2.1 `template<typename _Ptr> using std::pointer_traits< _Ptr >::difference_type = __detected_or_t<ptrdiff_t, __difference_type, _Ptr>`

The type used to represent the difference between two pointers.

Definition at line 100 of file ptr_traits.h.

5.938.2.2 `template<typename _Ptr> using std::pointer_traits< _Ptr >::element_type = __detected_or_t<__get_first_arg_t, __element_type, _Ptr>`

The type pointed to.

Definition at line 96 of file ptr_traits.h.

5.938.2.3 `template<typename _Ptr> using std::pointer_traits< _Ptr >::pointer = _Ptr`

The pointer type.

Definition at line 92 of file ptr_traits.h.

5.938.2.4 `template<typename _Ptr> template<typename _Up > using std::pointer_traits< _Ptr >::rebind =
__detected_or_t<__replace_first_arg_t, __rebind, _Ptr, _Up>`

A pointer to a different type.

Definition at line 105 of file `ptr_traits.h`.

The documentation for this struct was generated from the following file:

- [ptr_traits.h](#)

5.939 `std::pointer_traits< _Tp * >` Struct Template Reference

Public Types

- typedef ptrdiff_t [difference_type](#)
- typedef _Tp [element_type](#)
- typedef _Tp * [pointer](#)
- template<typename _Up >
using **rebind** = _Up *

Static Public Member Functions

- static [pointer pointer_to](#) (__make_not_void< [element_type](#) > &__r) noexcept

5.939.1 Detailed Description

```
template<typename _Tp>
struct std::pointer_traits< _Tp * >
```

Partial specialization for built-in pointers.

Definition at line 122 of file `ptr_traits.h`.

5.939.2 Member Typedef Documentation

5.939.2.1 `template<typename _Tp > typedef ptrdiff_t std::pointer_traits< _Tp * >::difference_type`

Type used to represent the difference between two pointers.

Definition at line 129 of file `ptr_traits.h`.

5.939.2.2 `template<typename _Tp> typedef _Tp std::pointer_traits<_Tp*>::element_type`

The type pointed to.

Definition at line 127 of file `ptr_traits.h`.

5.939.2.3 `template<typename _Tp> typedef _Tp* std::pointer_traits<_Tp*>::pointer`

The pointer type.



Definition at line 125 of file `ptr_traits.h`.

5.939.3 Member Function Documentation

5.939.3.1 `template<typename _Tp> static pointer std::pointer_traits<_Tp*>::pointer_to (__make_not_void<element_type> &__r) [inline], [static], [noexcept]`

Obtain a pointer to an object.

Parameters

	A reference to an object of type <code>element_type</code>
<code>__r</code>	
	
<code>__r</code>	
<code>r</code>	

Returns

`addressof(__r)`

Definition at line 140 of file `ptr_traits.h`.

References `std::addressof()`.

The documentation for this struct was generated from the following file:

- [ptr_traits.h](#)

5.940 `std::poisson_distribution<_IntType>` Class Template Reference

Classes

- struct [param_type](#)

Public Types

- typedef `_IntType` [result_type](#)

Public Member Functions

- **poisson_distribution** (double __mean=1.0)
- **poisson_distribution** (const [param_type](#) &__p)
- template<typename `_ForwardIterator` , typename `_UniformRandomNumberGenerator` >
void **generate** (`_ForwardIterator` __f, `_ForwardIterator` __t, `_UniformRandomNumberGenerator` &__urng)
- template<typename `_ForwardIterator` , typename `_UniformRandomNumberGenerator` >
void **generate** (`_ForwardIterator` __f, `_ForwardIterator` __t, `_UniformRandomNumberGenerator` &__urng, const [param_type](#) &__p)
- template<typename `_UniformRandomNumberGenerator` >
void **generate** ([result_type](#) *__f, [result_type](#) *__t, `_UniformRandomNumberGenerator` &__urng, const [param_type](#) &__p)
- [result_type](#) **max** () const
- double **mean** () const
- [result_type](#) **min** () const
- template<typename `_UniformRandomNumberGenerator` >
[result_type](#) **operator()** (`_UniformRandomNumberGenerator` &__urng)
- template<typename `_UniformRandomNumberGenerator` >
[result_type](#) **operator()** (`_UniformRandomNumberGenerator` &__urng, const [param_type](#) &__p)
- [param_type](#) **param** () const
- void **param** (const [param_type](#) &__param)
- void **reset** ()

Friends

- template<typename `_IntType1` , typename `_CharT` , typename `_Traits` >
`std::basic_ostream`< `_CharT`, `_Traits` > & **operator<<** (`std::basic_ostream`< `_CharT`, `_Traits` > &__os, const `std::poisson_distribution`< `_IntType1` > &__x)
- bool **operator==** (const `poisson_distribution` &__d1, const `poisson_distribution` &__d2)
- template<typename `_IntType1` , typename `_CharT` , typename `_Traits` >
`std::basic_istream`< `_CharT`, `_Traits` > & **operator>>** (`std::basic_istream`< `_CharT`, `_Traits` > &__is, `std::poisson_distribution`< `_IntType1` > &__x)

5.940.1 Detailed Description

```
template<typename _IntType = int>
class std::poisson_distribution< _IntType >
```

A discrete Poisson random number distribution.

The formula for the Poisson probability density function is $p(i|\mu) = \frac{\mu^i}{i!} e^{-\mu}$ where μ is the parameter of the distribution.

Definition at line 4265 of file random.h.

5.940.2 Member Typedef Documentation

5.940.2.1 `template<typename _IntType = int> typedef _IntType std::poisson_distribution< _IntType >::result_type`

The type of the range of the distribution.

Definition at line 4268 of file random.h.

5.940.3 Member Function Documentation

5.940.3.1 `template<typename _IntType = int> result_type std::poisson_distribution< _IntType >::max () const` [inline]

Returns the least upper bound value of the distribution.

Definition at line 4359 of file random.h.

References `std::numeric_limits< _Tp >::max()`.

5.940.3.2 `template<typename _IntType = int> double std::poisson_distribution< _IntType >::mean () const` [inline]

Returns the distribution parameter `mean`.

Definition at line 4330 of file random.h.

5.940.3.3 `template<typename _IntType = int> result_type std::poisson_distribution< _IntType >::min () const` [inline]

Returns the greatest lower bound value of the distribution.

Definition at line 4352 of file random.h.

5.940.3.4 `template<typename _IntType = int> template<typename _UniformRandomNumberGenerator > result_type` `std::poisson_distribution< _IntType >::operator() (_UniformRandomNumberGenerator & __urng)` [inline]

Generating functions.

Definition at line 4367 of file random.h.

Referenced by `std::negative_binomial_distribution< _IntType >::operator()()`.

5.940.3.5 `template<typename _IntType > template<typename _UniformRandomNumberGenerator > poisson_distribution<_IntType >::result_type std::poisson_distribution<_IntType >::operator() (_UniformRandomNumberGenerator & __urng, const param_type & __param)`

A rejection algorithm when mean ≥ 12 and a simple method based upon the multiplication of uniform random variates otherwise. NB: The former is available only if `_GLIBCXX_USE_C99_MATH_TR1` is defined.

Reference: Devroye, L. Non-Uniform Random Variates Generation. Springer-Verlag, New York, 1986, Ch. X, Sects. 3.3 & 3.4 (+ Errata!).

Definition at line 1284 of file `bits/random.tcc`.

References `std::abs()`, `std::numeric_limits<_Tp >::epsilon()`, `std::exp()`, `std::ios_base::flags()`, `std::left()`, `std::log()`, `std::numeric_limits<_Tp >::max()`, `std::binomial_distribution<_IntType >::operator()()`, `std::__detail::operator>>()`, `std::poisson_distribution<_IntType >::param()`, `std::scientific()`, `std::skipws()`, and `std::sqrt()`.

5.940.3.6 `template<typename _IntType = int> param_type std::poisson_distribution<_IntType >::param () const [inline]`

Returns the parameter set of the distribution.

Definition at line 4337 of file `random.h`.

Referenced by `std::poisson_distribution<_IntType >::operator()()`.

5.940.3.7 `template<typename _IntType = int> void std::poisson_distribution<_IntType >::param (const param_type & __param) [inline]`

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

Definition at line 4345 of file `random.h`.

5.940.3.8 `template<typename _IntType = int> void std::poisson_distribution<_IntType >::reset () [inline]`

Resets the distribution state.

Definition at line 4323 of file `random.h`.

5.940.4 Friends And Related Function Documentation

5.940.4.1 `template<typename _IntType = int> template<typename _IntType1, typename _CharT, typename _Traits > std::basic_ostream<_CharT, _Traits>& operator<< (std::basic_ostream<_CharT, _Traits> & __os, const std::poisson_distribution<_IntType1> & __x) [friend]`

Inserts a `poisson_distribution` random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A poisson_distribution random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

5.940.4.2 `template<typename _IntType = int> bool operator==(const poisson_distribution<_IntType> &__d1, const poisson_distribution<_IntType> &__d2) [friend]`

Return true if two Poisson distributions have the same parameters and the sequences that would be generated are equal.

Definition at line 4403 of file random.h.

5.940.4.3 `template<typename _IntType = int> template<typename _IntType1, typename _CharT, typename _Traits> std::basic_istream<_CharT, _Traits>& operator>> (std::basic_istream<_CharT, _Traits> &__is, std::poisson_distribution<_IntType1> &__x) [friend]`

Extracts a poisson_distribution random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A poisson_distribution random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.941 std::poisson_distribution<_IntType>::param_type Struct Reference

Public Types

- typedef [poisson_distribution<_IntType>](#) **distribution_type**

Public Member Functions

- **param_type** (double __mean=1.0)
- double **mean** () const

Friends

- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)
- class **poisson_distribution**< [_IntType](#) >

5.941.1 Detailed Description

```
template<typename _IntType = int>
struct std::poisson_distribution< _IntType >::param_type
```

Parameter type.

Definition at line 4274 of file random.h.

The documentation for this struct was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.942 **std::priority_queue< _Tp, _Sequence, _Compare > Class Template Reference**

Public Types

- typedef [_Sequence::const_reference](#) **const_reference**
- typedef [_Sequence](#) **container_type**
- typedef [_Sequence::reference](#) **reference**
- typedef [_Sequence::size_type](#) **size_type**
- typedef [_Sequence::value_type](#) **value_type**

Public Member Functions

- [priority_queue](#) (const _Compare &__x, const _Sequence &__s)
- **priority_queue** (const _Compare &__x=_Compare(), _Sequence &&__s=_Sequence())
- template<typename _Alloc, typename _Requires = _Uses<_Alloc>>
priority_queue (const _Alloc &__a)
- template<typename _Alloc, typename _Requires = _Uses<_Alloc>>
priority_queue (const _Compare &__x, const _Alloc &__a)
- template<typename _Alloc, typename _Requires = _Uses<_Alloc>>
priority_queue (const _Compare &__x, const _Sequence &__c, const _Alloc &__a)
- template<typename _Alloc, typename _Requires = _Uses<_Alloc>>
priority_queue (const _Compare &__x, _Sequence &&__c, const _Alloc &__a)
- template<typename _Alloc, typename _Requires = _Uses<_Alloc>>
priority_queue (const [priority_queue](#) &__q, const _Alloc &__a)
- template<typename _Alloc, typename _Requires = _Uses<_Alloc>>
priority_queue ([priority_queue](#) &&__q, const _Alloc &__a)
- template<typename _InputIterator >
[priority_queue](#) (_InputIterator __first, _InputIterator __last, const _Compare &__x, const _Sequence &__s)
- template<typename _InputIterator >
priority_queue (_InputIterator __first, _InputIterator __last, const _Compare &__x=_Compare(), _Sequence &&__s=_Sequence())
- template<typename... _Args>
void **emplace** (_Args &&...__args)
- bool [empty](#) () const
- void [pop](#) ()
- void [push](#) (const value_type &__x)
- void **push** (value_type &&__x)
- size_type [size](#) () const
- void **swap** ([priority_queue](#) &__pq) noexcept(__is_nothrow_swappable< _Tp >::value &&__is_nothrow_swappable< _Compare >::value)
- const_reference [top](#) () const

Protected Attributes

- _Sequence **c**
- _Compare **comp**

5.942.1 Detailed Description

```
template<typename _Tp, typename _Sequence = vector<_Tp>, typename _Compare = less<typename _Sequence::value_type>>
class std::priority_queue< _Tp, _Sequence, _Compare >
```

A standard container automatically sorting its contents.

Template Parameters

<code>_Tp</code>	Type of element.
<code>_Sequence</code>	Type of underlying sequence, defaults to <code>vector<_Tp></code> .
<code>_Compare</code>	Comparison function object type, defaults to <code>less<_Sequence::value_type></code> .

This is not a true container, but an *adaptor*. It holds another container, and provides a wrapper interface to that container. The wrapper is what enforces priority-based sorting and queue behavior. Very few of the standard container/sequence interface requirements are met (e.g., iterators).

The second template parameter defines the type of the underlying sequence/container. It defaults to `std::vector`, but it can be any type that supports `front()`, `push_back`, `pop_back`, and random-access iterators, such as `std::deque` or an appropriate user-defined type.

The third template parameter supplies the means of making priority comparisons. It defaults to `less<value_type>` but can be anything defining a strict weak ordering.

Members not found in *normal* containers are `container_type`, which is a typedef for the second Sequence parameter, and `push`, `pop`, and `top`, which are standard queue operations.

Note

No equality/comparison operators are provided for `priority_queue`.

Sorting of the elements takes place as they are added to, and removed from, the `priority_queue` using the `priority_queue`'s member functions. If you access the elements by other means, and change their data such that the sorting order would be different, the `priority_queue` will not re-sort the elements for you. (How could it know to do so?)

Definition at line 397 of file `stl_queue.h`.

5.942.2 Constructor & Destructor Documentation

5.942.2.1 `template<typename _Tp, typename _Sequence = vector<_Tp>, typename _Compare = less<typename _Sequence::value_type>> std::priority_queue<_Tp, _Sequence, _Compare>::priority_queue (const _Compare &__x, const _Sequence &__s) [inline], [explicit]`

Default constructor creates no elements.

Definition at line 438 of file `stl_queue.h`.

References `std::queue<_Tp, _Sequence>::c`, and `std::make_heap()`.

5.942.2.2 `template<typename _Tp, typename _Sequence = vector<_Tp>, typename _Compare = less<typename _Sequence::value_type>> template<typename _InputIterator> std::priority_queue<_Tp, _Sequence, _Compare>::priority_queue (_InputIterator __first, _InputIterator __last, const _Compare &__x, const _Sequence &__s) [inline]`

Builds a queue from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__x</code>	A comparison functor describing a strict weak ordering.
<code>__s</code>	An initial sequence with which to start.

Begins by copying __s, inserting a copy of the elements from [first,last) into the copy of __s, then ordering the copy according to __x.

For more information on function objects, see the documentation on [functor base classes](#).

Definition at line 504 of file stl_queue.h.

References std::queue< _Tp, _Sequence >::c, and std::make_heap().

5.942.3 Member Function Documentation

5.942.3.1 `template<typename _Tp, typename _Sequence = vector<_Tp>, typename _Compare = less<typename _Sequence::value_type>> bool std::priority_queue<_Tp, _Sequence, _Compare >::empty () const [inline]`

Returns true if the queue is empty.

Definition at line 530 of file stl_queue.h.

Referenced by __gnu_parallel::multiseq_partition(), and __gnu_parallel::multiseq_selection().

5.942.3.2 `template<typename _Tp, typename _Sequence = vector<_Tp>, typename _Compare = less<typename _Sequence::value_type>> void std::priority_queue<_Tp, _Sequence, _Compare >::pop () [inline]`

Removes first element.

This is a typical queue operation. It shrinks the queue by one. The time complexity of the operation depends on the underlying sequence.

Note that no data is returned, and if the first element's data is needed, it should be retrieved before pop() is called.

Definition at line 593 of file stl_queue.h.

References std::pop_heap().

Referenced by __gnu_parallel::multiseq_partition(), and __gnu_parallel::multiseq_selection().

5.942.3.3 `template<typename _Tp, typename _Sequence = vector<_Tp>, typename _Compare = less<typename _Sequence::value_type>> void std::priority_queue<_Tp, _Sequence, _Compare >::push (const value_type & __x) [inline]`

Add data to the queue.

Parameters

<code>__x</code>	Data to be added.
------------------	-------------------

This is a typical queue operation. The time complexity of the operation depends on the underlying sequence.

Definition at line 558 of file `stl_queue.h`.

References `std::queue<_Tp, _Sequence >::push()`, and `std::push_heap()`.

Referenced by `__gnu_parallel::multiseq_partition()`, and `__gnu_parallel::multiseq_selection()`.

5.942.3.4 `template<typename _Tp, typename _Sequence = vector<_Tp>, typename _Compare = less<typename
_Sequence::value_type>> size_type std::priority_queue<_Tp, _Sequence, _Compare >::size () const
[inline]`

Returns the number of elements in the queue.

Definition at line 535 of file `stl_queue.h`.

5.942.3.5 `template<typename _Tp, typename _Sequence = vector<_Tp>, typename _Compare = less<typename
_Sequence::value_type>> const_reference std::priority_queue<_Tp, _Sequence, _Compare >::top () const
[inline]`

Returns a read-only (constant) reference to the data at the first element of the queue.

Definition at line 543 of file `stl_queue.h`.

Referenced by `__gnu_parallel::multiseq_partition()`, and `__gnu_parallel::multiseq_selection()`.

The documentation for this class was generated from the following file:

- [stl_queue.h](#)

5.943 `std::promise<_Res >` Class Template Reference

Public Member Functions

- **promise** ([promise](#) &&__rhs) noexcept
- `template<typename _Allocator >`
promise ([allocator_arg_t](#), const _Allocator &__a)
- `template<typename _Allocator >`
promise ([allocator_arg_t](#), const _Allocator &, [promise](#) &&__rhs)
- **promise** (const [promise](#) &)=delete
- **future**<_Res > **get_future** ()
- [promise](#) & **operator=** ([promise](#) &&__rhs) noexcept
- [promise](#) & **operator=** (const [promise](#) &)=delete
- void **set_exception** (exception_ptr __p)
- void **set_exception_at_thread_exit** (exception_ptr __p)
- void **set_value** (const _Res &__r)
- void **set_value** (_Res &&__r)
- void **set_value_at_thread_exit** (const _Res &__r)
- void **set_value_at_thread_exit** (_Res &&__r)
- void **swap** ([promise](#) &&__rhs) noexcept

Friends

- `template<typename , typename >`
`class _State::_Setter`

5.943.1 Detailed Description

```
template<typename _Res>
class std::promise< _Res >
```

Primary template for promise.

Definition at line 124 of file future.

The documentation for this class was generated from the following file:

- [future](#)

5.944 std::promise< _Res & > Class Template Reference

Public Member Functions

- **promise** ([promise](#) &&__rhs) noexcept
- `template<typename _Allocator >`
promise ([allocator_arg_t](#), const _Allocator &__a)
- `template<typename _Allocator >`
promise ([allocator_arg_t](#), const _Allocator &, [promise](#) &&__rhs)
- **promise** (const [promise](#) &)=delete
- [future](#)< _Res & > **get_future** ()
- [promise](#) & **operator=** ([promise](#) &&__rhs) noexcept
- [promise](#) & **operator=** (const [promise](#) &)=delete
- void **set_exception** (exception_ptr __p)
- void **set_exception_at_thread_exit** (exception_ptr __p)
- void **set_value** (_Res &__r)
- void **set_value_at_thread_exit** (_Res &__r)
- void **swap** ([promise](#) &&__rhs) noexcept

Friends

- `template<typename , typename >`
`class _State::_Setter`

5.944.1 Detailed Description

```
template<typename _Res>
class std::promise< _Res & >
```

Partial specialization for promise<R&>

Definition at line 1121 of file future.

The documentation for this class was generated from the following file:

- [future](#)

5.945 std::promise< void > Class Template Reference

Public Member Functions

- **promise** ([promise](#) &&__rhs) noexcept
- template<typename _Allocator >
promise ([allocator_arg_t](#), const _Allocator &__a)
- template<typename _Allocator >
promise ([allocator_arg_t](#), const _Allocator &, [promise](#) &&__rhs)
- **promise** (const [promise](#) &)=delete
- [future](#)< void > **get_future** ()
- [promise](#) & **operator=** ([promise](#) &&__rhs) noexcept
- [promise](#) & **operator=** (const [promise](#) &)=delete
- void **set_exception** (exception_ptr __p)
- void **set_exception_at_thread_exit** (exception_ptr __p)
- void **set_value** ()
- void **set_value_at_thread_exit** ()
- void **swap** ([promise](#) &__rhs) noexcept

Friends

- template<typename , typename >
class **_State::_Setter**

5.945.1 Detailed Description

```
template<>
class std::promise< void >
```

Explicit specialization for promise<void>

Definition at line 1210 of file future.

The documentation for this class was generated from the following file:

- [future](#)

5.946 std::queue< _Tp, _Sequence > Class Template Reference

Public Types

- typedef _Sequence::const_reference **const_reference**
- typedef _Sequence **container_type**
- typedef _Sequence::reference **reference**
- typedef _Sequence::size_type **size_type**
- typedef _Sequence::value_type **value_type**

Public Member Functions

- [queue](#) (const _Sequence &__c)
- [queue](#) (_Sequence &&__c=_Sequence())
- template<typename _Alloc, typename _Requires = _Uses<_Alloc>>
[queue](#) (const _Alloc &__a)
- template<typename _Alloc, typename _Requires = _Uses<_Alloc>>
[queue](#) (const _Sequence &__c, const _Alloc &__a)
- template<typename _Alloc, typename _Requires = _Uses<_Alloc>>
[queue](#) (_Sequence &&__c, const _Alloc &__a)
- template<typename _Alloc, typename _Requires = _Uses<_Alloc>>
[queue](#) (const [queue](#) &__q, const _Alloc &__a)
- template<typename _Alloc, typename _Requires = _Uses<_Alloc>>
[queue](#) ([queue](#) &&__q, const _Alloc &__a)
- reference [back](#) ()
- const_reference [back](#) () const
- template<typename... _Args>
void [emplace](#) (_Args &&... __args)
- bool [empty](#) () const
- reference [front](#) ()
- const_reference [front](#) () const
- void [pop](#) ()
- void [push](#) (const value_type &__x)
- void [push](#) (value_type &&__x)
- size_type [size](#) () const
- void [swap](#) ([queue](#) &__q) noexcept(__is_nothrow_swappable<_Tp>::value)

Protected Attributes

- _Sequence [c](#)

Friends

- template<typename _Tp1, typename _Seq1 >
bool [operator](#)< (const [queue](#)< _Tp1, _Seq1 > &, const [queue](#)< _Tp1, _Seq1 > &)
- template<typename _Tp1, typename _Seq1 >
bool [operator](#)== (const [queue](#)< _Tp1, _Seq1 > &, const [queue](#)< _Tp1, _Seq1 > &)

5.946.1 Detailed Description

```
template<typename _Tp, typename _Sequence = deque<_Tp>>
class std::queue< _Tp, _Sequence >
```

A standard container giving FIFO behavior.

Template Parameters

<code>_Tp</code>	Type of element.
<code>_Sequence</code>	Type of underlying sequence, defaults to <code>deque<_Tp></code> .

Meets many of the requirements of a `container`, but does not define anything to do with iterators. Very few of the other standard container interfaces are defined.

This is not a true container, but an *adaptor*. It holds another container, and provides a wrapper interface to that container. The wrapper is what enforces strict first-in-first-out queue behavior.

The second template parameter defines the type of the underlying sequence/container. It defaults to `std::deque`, but it can be any type that supports `front`, `back`, `push_back`, and `pop_front`, such as `std::list` or an appropriate user-defined type.

Members not found in *normal* containers are `container_type`, which is a typedef for the second Sequence parameter, and `push` and `pop`, which are standard queue/FIFO operations.

Definition at line 96 of file `stl_queue.h`.

5.946.2 Constructor & Destructor Documentation

5.946.2.1 `template<typename _Tp, typename _Sequence = deque<_Tp>> std::queue<_Tp, _Sequence>::queue (const _Sequence &__c) [inline], [explicit]`

Default constructor creates no elements.

Definition at line 147 of file `stl_queue.h`.

References `std::queue<_Tp, _Sequence>::c`.

5.946.3 Member Function Documentation

5.946.3.1 `template<typename _Tp, typename _Sequence = deque<_Tp>> reference std::queue<_Tp, _Sequence>::back () [inline]`

Returns a read/write reference to the data at the last element of the queue.

Definition at line 215 of file `stl_queue.h`.

5.946.3.2 `template<typename _Tp, typename _Sequence = deque<_Tp>> const_reference std::queue<_Tp, _Sequence>::back () const [inline]`

Returns a read-only (constant) reference to the data at the last element of the queue.

Definition at line 226 of file `stl_queue.h`.

5.946.3.3 `template<typename _Tp, typename _Sequence = deque<_Tp>> bool std::queue<_Tp, _Sequence>::empty ()`
`const [inline]`

Returns true if the queue is empty.

Definition at line 180 of file `stl_queue.h`.

5.946.3.4 `template<typename _Tp, typename _Sequence = deque<_Tp>> reference std::queue<_Tp, _Sequence>::front ()`
`[inline]`

Returns a read/write reference to the data at the first element of the queue.

Definition at line 193 of file `stl_queue.h`.

5.946.3.5 `template<typename _Tp, typename _Sequence = deque<_Tp>> const_reference std::queue<_Tp, _Sequence>::front () const` `[inline]`

Returns a read-only (constant) reference to the data at the first element of the queue.

Definition at line 204 of file `stl_queue.h`.

5.946.3.6 `template<typename _Tp, typename _Sequence = deque<_Tp>> void std::queue<_Tp, _Sequence>::pop ()`
`[inline]`

Removes first element.

This is a typical queue operation. It shrinks the queue by one. The time complexity of the operation depends on the underlying sequence.

Note that no data is returned, and if the first element's data is needed, it should be retrieved before `pop()` is called.

Definition at line 268 of file `stl_queue.h`.

References `std::queue<_Tp, _Sequence>::c`.

5.946.3.7 `template<typename _Tp, typename _Sequence = deque<_Tp>> void std::queue<_Tp, _Sequence>::push (const value_type &__x)` `[inline]`

Add data to the end of the queue.

Parameters

<code>__x</code>	Data to be added.
------------------	-------------------

This is a typical queue operation. The function creates an element at the end of the queue and assigns the given data to it. The time complexity of the operation depends on the underlying sequence.

Definition at line 242 of file `stl_queue.h`.

Referenced by `std::priority_queue< _Tp, _Sequence, _Compare >::push()`.

5.946.3.8 `template<typename _Tp, typename _Sequence = deque<_Tp>> size_type std::queue< _Tp, _Sequence >::size ()`
`const [inline]`

Returns the number of elements in the queue.

Definition at line 185 of file `stl_queue.h`.

5.946.4 Member Data Documentation

5.946.4.1 `template<typename _Tp, typename _Sequence = deque<_Tp>> _Sequence std::queue< _Tp, _Sequence >::c`
`[protected]`

'c' is the underlying container. Maintainers wondering why this isn't uglified as per style guidelines should note that this name is specified in the standard, [23.2.3.1]. (Why? Presumably for the same reason that it's protected instead of private: to allow derivation. But none of the other containers allow for derivation. Odd.)

Definition at line 135 of file `stl_queue.h`.

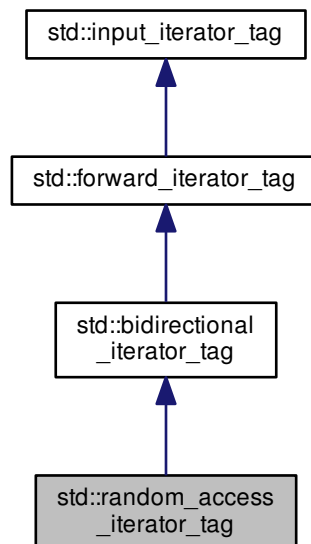
Referenced by `std::operator<()`, `std::operator==()`, `std::queue< _Tp, _Sequence >::pop()`, `std::priority_queue< _Tp, _Sequence, _Compare >::priority_queue()`, and `std::queue< _Tp, _Sequence >::queue()`.

The documentation for this class was generated from the following file:

- [stl_queue.h](#)

5.947 `std::random_access_iterator_tag` Struct Reference

Inheritance diagram for `std::random_access_iterator_tag`:



5.947.1 Detailed Description

Random-access iterators support a superset of bidirectional iterator operations.

Definition at line 103 of file `stl_iterator_base_types.h`.

The documentation for this struct was generated from the following file:

- [stl_iterator_base_types.h](#)

5.948 `std::random_device` Class Reference

Public Types

- typedef unsigned int [result_type](#)

Public Member Functions

- **random_device** (const [std::string](#) &__token="mt19937")
- **random_device** (const [random_device](#) &)=delete
- double **entropy** () const noexcept
- [result_type](#) **operator**() ()
- void **operator=** (const [random_device](#) &)=delete

Static Public Member Functions

- static constexpr [result_type](#) **max** ()
- static constexpr [result_type](#) **min** ()

5.948.1 Detailed Description

A standard interface to a platform-specific non-deterministic random number generator (if any are available).

Definition at line 1567 of file `random.h`.

5.948.2 Member Typedef Documentation

5.948.2.1 typedef unsigned int `std::random_device::result_type`

The type of the generated random value.

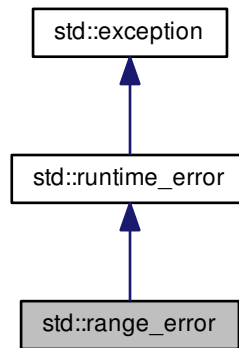
Definition at line 1571 of file `random.h`.

The documentation for this class was generated from the following file:

- [random.h](#)

5.949 `std::range_error` Class Reference

Inheritance diagram for `std::range_error`:



Public Member Functions

- **`range_error`** (const [string](#) &__arg) `_GLIBCXX_TXN_SAFE`
- **`range_error`** (const char *) `_GLIBCXX_TXN_SAFE`
- virtual const char * [what](#) () const `_GLIBCXX_TXN_SAFE_DYN noexcept`

5.949.1 Detailed Description

Thrown to indicate range errors in internal computations.

Definition at line 230 of file `stdexcept`.

5.949.2 Member Function Documentation

5.949.2.1 virtual const char* `std::runtime_error::what` () const `[virtual], [noexcept], [inherited]`

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

The documentation for this class was generated from the following file:

- [stdexcept](#)

5.950 `std::ratio< _Num, _Den >` Struct Template Reference

Public Types

- typedef `ratio< num, den >` **type**

Static Public Attributes

- static constexpr intmax_t **den**
- static constexpr intmax_t **num**

5.950.1 Detailed Description

```
template<intmax_t _Num, intmax_t _Den = 1>
struct std::ratio< _Num, _Den >
```

Provides compile-time rational arithmetic.

This class template represents any finite rational number with a numerator and denominator representable by compile-time constants of type `intmax_t`. The ratio is simplified when instantiated.

For example:

```
1 std::ratio<7,-21>::num == -1;
2 std::ratio<7,-21>::den == 3;
```

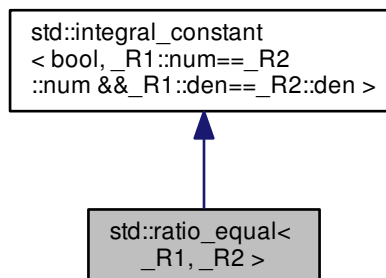
Definition at line 263 of file `ratio`.

The documentation for this struct was generated from the following file:

- `ratio`

5.951 `std::ratio_equal< _R1, _R2 >` Struct Template Reference

Inheritance diagram for `std::ratio_equal< _R1, _R2 >`:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr bool **value**

5.951.1 Detailed Description

```
template<typename _R1, typename _R2>
struct std::ratio_equal< _R1, _R2 >
```

ratio_equal

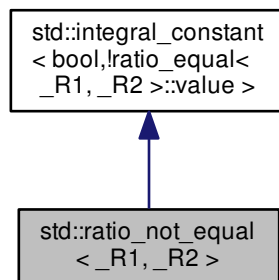
Definition at line 340 of file ratio.

The documentation for this struct was generated from the following file:

- [ratio](#)

5.952 std::ratio_not_equal< _R1, _R2 > Struct Template Reference

Inheritance diagram for std::ratio_not_equal< _R1, _R2 >:



Public Types

- typedef [integral_constant](#)< bool, __v > **type**
- typedef bool **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr bool **value**

5.952.1 Detailed Description

```
template<typename _R1, typename _R2>
struct std::ratio_not_equal< _R1, _R2 >
```

ratio_not_equal

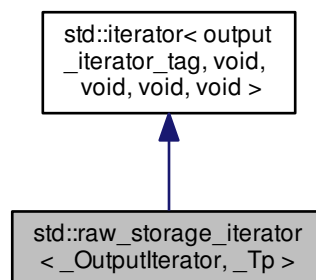
Definition at line 346 of file ratio.

The documentation for this struct was generated from the following file:

- [ratio](#)

5.953 std::raw_storage_iterator< _OutputIterator, _Tp > Class Template Reference

Inheritance diagram for std::raw_storage_iterator< _OutputIterator, _Tp >:



Public Types

- typedef void [difference_type](#)
- typedef [output_iterator_tag](#) [iterator_category](#)
- typedef void [pointer](#)
- typedef void [reference](#)
- typedef void [value_type](#)

Public Member Functions

- **raw_storage_iterator** ([_OutputIterator](#) __x)
- [_OutputIterator](#) **base** () const
- [raw_storage_iterator](#) & **operator*** ()
- [raw_storage_iterator](#) & **operator++** ()
- [raw_storage_iterator](#) **operator++** (int)
- [raw_storage_iterator](#) & **operator=** (const [_Tp](#) &__element)
- [raw_storage_iterator](#) & **operator=** ([_Tp](#) &&__element)

Protected Attributes

- [_OutputIterator](#) **_M_iter**

5.953.1 Detailed Description

```
template<class _OutputIterator, class _Tp>
class std::raw_storage_iterator< \_OutputIterator, \_Tp >
```

This iterator class lets algorithms store their results into uninitialized memory.

Definition at line 68 of file `stl_raw_storage_iter.h`.

5.953.2 Member Typedef Documentation

5.953.2.1 typedef void `std::iterator< output_iterator_tag, void, void, void, void >::difference_type` [inherited]

Distance between iterators is represented as this type.

Definition at line 125 of file `stl_iterator_base_types.h`.

5.953.2.2 typedef [output_iterator_tag](#) `std::iterator< output_iterator_tag, void, void, void, void >::iterator_category` [inherited]

One of the [tag types](#).

Definition at line 121 of file `stl_iterator_base_types.h`.

5.953.2.3 `typedef void std::iterator< output_iterator_tag, void, void, void, void >::pointer` [inherited]

This type represents a pointer-to-value_type.

Definition at line 127 of file stl_iterator_base_types.h.

5.953.2.4 `typedef void std::iterator< output_iterator_tag, void, void, void, void >::reference` [inherited]

This type represents a reference-to-value_type.

Definition at line 129 of file stl_iterator_base_types.h.

5.953.2.5 `typedef void std::iterator< output_iterator_tag, void, void, void, void >::value_type` [inherited]

The type "pointed to" by the iterator.

Definition at line 123 of file stl_iterator_base_types.h.

The documentation for this class was generated from the following file:

- [stl_raw_storage_iter.h](#)

5.954 std::recursive_mutex Class Reference

Inherits std::__recursive_mutex_base.

Public Types

- `typedef __native_type * native_handle_type`

Public Member Functions

- `recursive_mutex` (const [recursive_mutex](#) &)=delete
- `void lock` ()
- `native_handle_type native_handle` ()
- `recursive_mutex` & `operator=` (const [recursive_mutex](#) &)=delete
- `bool try_lock` () noexcept
- `void unlock` ()

Private Types

- `typedef __pthread_recursive_mutex_t __native_type`

Private Attributes

- `__native_type __M_mutex`

5.954.1 Detailed Description

The standard recursive mutex type.

Definition at line 91 of file `mutex`.

The documentation for this class was generated from the following file:

- [mutex](#)

5.955 `std::recursive_timed_mutex` Class Reference

Public Member Functions

- **`recursive_timed_mutex`** (const [recursive_timed_mutex](#) &)=delete
- void **`lock`** ()
- [recursive_timed_mutex](#) & **`operator=`** (const [recursive_timed_mutex](#) &)=delete
- bool **`try_lock`** ()
- template<typename `_Rep` , typename `_Period` >
bool **`try_lock_for`** (const [chrono::duration](#)< `_Rep`, `_Period` > &__rtime)
- template<typename `_Clock` , typename `_Duration` >
bool **`try_lock_until`** (const [chrono::time_point](#)< `_Clock`, `_Duration` > &__atime)
- void **`unlock`** ()

5.955.1 Detailed Description

`recursive_timed_mutex`

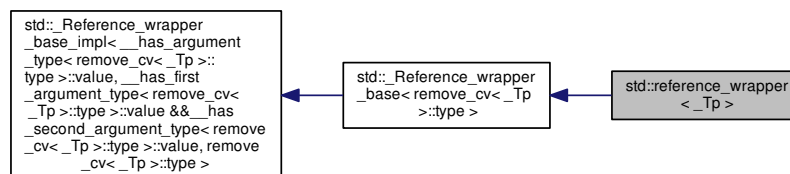
Definition at line 361 of file `mutex`.

The documentation for this class was generated from the following file:

- [mutex](#)

5.956 `std::reference_wrapper<_Tp>` Class Template Reference

Inheritance diagram for `std::reference_wrapper<_Tp>`:



Public Types

- typedef `_Tp` **type**

Public Member Functions

- **reference_wrapper** (`_Tp &__indata`) noexcept
- **reference_wrapper** (`_Tp &&__indata`)=delete
- **reference_wrapper** (const [reference_wrapper](#) &)=default
- `_Tp &` **get** () const noexcept
- **operator _Tp &** () const noexcept
- template<typename... `_Args`>
 result_of< `_Tp &(_Args &&...)`>::type **operator()** (`_Args &&...__args`) const
- [reference_wrapper](#) & **operator=** (const [reference_wrapper](#) &)=default

5.956.1 Detailed Description

```
template<typename _Tp>  
class std::reference_wrapper< _Tp >
```

Primary class template for `reference_wrapper`.

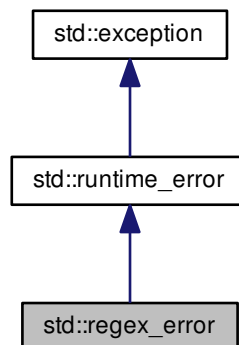
Definition at line 435 of file `functional`.

The documentation for this class was generated from the following file:

- [functional](#)

5.957 std::regex_error Class Reference

Inheritance diagram for `std::regex_error`:



Public Member Functions

- [regex_error](#) ([regex_constants::error_type](#) __ecode)
- [regex_constants::error_type](#) code () const
- virtual const char * [what](#) () const _GLIBCXX_TXN_SAFE_DYN noexcept

Friends

- void [__throw_regex_error](#) ([regex_constants::error_type](#), const char *)

5.957.1 Detailed Description

A regular expression exception class.

The regular expression library throws objects of this class on error.

Definition at line 135 of file [regex_error.h](#).

5.957.2 Constructor & Destructor Documentation

5.957.2.1 `std::regex_error::regex_error (regex_constants::error_type __ecode)` [\[explicit\]](#)

Constructs a `regex_error` object.

Parameters

<code>__ecode</code>	the regex error code.
----------------------	-----------------------

5.957.3 Member Function Documentation

5.957.3.1 `regex_constants::error_type std::regex_error::code () const` [\[inline\]](#)

Gets the regex error code.

Returns

the regex error code.

Definition at line 156 of file [regex_error.h](#).

5.957.3.2 `virtual const char* std::runtime_error::what () const` [\[virtual\]](#), [\[noexcept\]](#), [\[inherited\]](#)

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

The documentation for this class was generated from the following file:

- [regex_error.h](#)

5.958 std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits > Class Template Reference

Public Types

- typedef std::ptrdiff_t **difference_type**
- typedef [std::forward_iterator_tag](#) **iterator_category**
- typedef const [value_type](#) * **pointer**
- typedef const [value_type](#) & **reference**
- typedef [basic_regex](#)< _Ch_type, _Rx_traits > **regex_type**
- typedef [match_results](#)< _Bi_iter > **value_type**

Public Member Functions

- [regex_iterator](#) ()
- [regex_iterator](#) (_Bi_iter __a, _Bi_iter __b, const [regex_type](#) &__re, [regex_constants::match_flag_type](#) __m= [regex_constants::match_default](#))
- **regex_iterator** (_Bi_iter, _Bi_iter, const [regex_type](#) &&, [regex_constants::match_flag_type](#)=[regex_constants::match_default](#))=delete
- [regex_iterator](#) (const [regex_iterator](#) &__rhs)=default
- bool [operator!=](#) (const [regex_iterator](#) &__rhs) const
- const [value_type](#) & [operator*](#) () const
- [regex_iterator](#) & [operator++](#) ()
- [regex_iterator](#) [operator++](#) (int)
- const [value_type](#) * [operator->](#) () const
- [regex_iterator](#) & [operator=](#) (const [regex_iterator](#) &__rhs)=default
- bool [operator==](#) (const [regex_iterator](#) &__rhs) const

5.958.1 Detailed Description

```
template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename _Rx_traits = regex_traits<_Ch_type>>
class std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >
```

An iterator adaptor that will provide repeated calls of `regex_search` over a range until no more matches remain.

Definition at line 2442 of file `regex.h`.

5.958.2 Constructor & Destructor Documentation

5.958.2.1 `template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename _Rx_traits = regex_traits<_Ch_type>> std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::regex_iterator ()` `[inline]`

Provides a singular iterator, useful for indicating one-past-the-end of a range.

Definition at line 2456 of file `regex.h`.

5.958.2.2 `template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename _Rx_traits = regex_traits<_Ch_type>> std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::regex_iterator (_Bi_iter __a, _Bi_iter __b, const regex_type & __re, regex_constants::match_flag_type __m = regex_constants::match_default)` `[inline]`

Constructs a `regex_iterator`...

Parameters

<code>_↔ _a</code>	[IN] The start of a text range to search.
<code>_↔ _b</code>	[IN] One-past-the-end of the text range to search.
<code>_↔ _re</code>	[IN] The regular expression to match.
<code>_↔ _m</code>	[IN] Policy flags for match rules.

Definition at line 2467 of file `regex.h`.

References `std::regex_constants::match_default`, and `std::regex_search()`.

```
5.958.2.3  template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename _Rx_traits
           = regex_traits<_Ch_type>> std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::regex_iterator ( const
           regex_iterator< _Bi_iter, _Ch_type, _Rx_traits > & __rhs ) [default]
```

Copy constructs a `regex_iterator`.

5.958.3 Member Function Documentation

```
5.958.3.1  template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename _Rx_traits
           = regex_traits<_Ch_type>> bool std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator!= ( const
           regex_iterator< _Bi_iter, _Ch_type, _Rx_traits > & __rhs ) const [inline]
```

Tests the inequivalence of two `regex` iterators.

Definition at line 2502 of file `regex.h`.

```
5.958.3.2  template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename _Rx_traits
           = regex_traits<_Ch_type>> const value_type& std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator* (
           ) const [inline]
```

Dereferences a `regex_iterator`.

Definition at line 2509 of file `regex.h`.

```
5.958.3.3  template<typename _Bi_iter, typename _Ch_type, typename _Rx_traits > regex_iterator< _Bi_iter, _Ch_type,
           _Rx_traits > & std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator++ ( )
```

Increments a `regex_iterator`.

Definition at line 512 of file `regex.tcc`.

References `std::regex_constants::match_continuous`, `std::regex_constants::match_not_null`, `std::regex_constants::↔
::match_prev_avail`, `std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator=()`, and `std::regex_search()`.

Referenced by `std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator==()`.

5.958.3.4 `template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename _Rx_traits = regex_traits<_Ch_type>> regex_iterator std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator++(int) [inline]`

Postincrements a `regex_iterator`.

Definition at line 2529 of file `regex.h`.

5.958.3.5 `template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename _Rx_traits = regex_traits<_Ch_type>> const value_type* std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator-> () const [inline]`

Selects a `regex_iterator` member.

Definition at line 2516 of file `regex.h`.

5.958.3.6 `template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename _Rx_traits = regex_traits<_Ch_type>> regex_iterator& std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator= (const regex_iterator< _Bi_iter, _Ch_type, _Rx_traits > & __rhs) [default]`

Assigns one `regex_iterator` to another.

5.958.3.7 `template<typename _Bi_iter, typename _Ch_type, typename _Rx_traits > bool std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator==(const regex_iterator< _Bi_iter, _Ch_type, _Rx_traits > & __rhs) const`

Tests the equivalence of two `regex` iterators.

Definition at line 497 of file `regex.tcc`.

References `std::match_results< _Bi_iter, _Alloc >::empty()`, and `std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator++()`.

Referenced by `std::regex_replace()`.

The documentation for this class was generated from the following files:

- [regex.h](#)
- [regex.tcc](#)

5.959 `std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >` Class Template Reference

Public Types

- typedef `std::ptrdiff_t` **difference_type**
- typedef `std::forward_iterator_tag` **iterator_category**
- typedef const `value_type` * **pointer**
- typedef const `value_type` & **reference**
- typedef `basic_regex< _Ch_type, _Rx_traits >` **regex_type**
- typedef `sub_match< _Bi_iter >` **value_type**

Public Member Functions

- `regex_token_iterator` ()
- `regex_token_iterator` (`_Bi_iter __a`, `_Bi_iter __b`, const `regex_type` &__re, int __submatch=0, `regex_constants::match_flag_type __m=regex_constants::match_default`)
- `regex_token_iterator` (`_Bi_iter __a`, `_Bi_iter __b`, const `regex_type` &__re, const `std::vector< int >` &__submatches, `regex_constants::match_flag_type __m=regex_constants::match_default`)
- `regex_token_iterator` (`_Bi_iter __a`, `_Bi_iter __b`, const `regex_type` &__re, `initializer_list< int >` __submatches, `regex_constants::match_flag_type __m=regex_constants::match_default`)
- `template<std::size_t _Nm>`
`regex_token_iterator` (`_Bi_iter __a`, `_Bi_iter __b`, const `regex_type` &__re, const `int(&__submatches)[_Nm]`, `regex_constants::match_flag_type __m=regex_constants::match_default`)
- `regex_token_iterator` (`_Bi_iter`, `_Bi_iter`, const `regex_type` &&, int=0, `regex_constants::match_flag_type=regex_constants::match_default`)=delete
- `regex_token_iterator` (`_Bi_iter`, `_Bi_iter`, const `regex_type` &&, const `std::vector< int >` &, `regex_constants::match_flag_type=regex_constants::match_default`)=delete
- `regex_token_iterator` (`_Bi_iter`, `_Bi_iter`, const `regex_type` &&, `initializer_list< int >`, `regex_constants::match_flag_type=regex_constants::match_default`)=delete
- `template<std::size_t N>`
`regex_token_iterator` (`_Bi_iter`, `_Bi_iter`, const `regex_type` &&, const `int(&)[N]`, `regex_constants::match_flag_type=regex_constants::match_default`)=delete
- `regex_token_iterator` (const `regex_token_iterator` &__rhs)
- `bool operator!=` (const `regex_token_iterator` &__rhs) const
- `const value_type & operator*` () const
- `regex_token_iterator & operator++` ()
- `regex_token_iterator operator++` (int)
- `const value_type * operator->` () const
- `regex_token_iterator & operator=` (const `regex_token_iterator` &__rhs)
- `bool operator==` (const `regex_token_iterator` &__rhs) const

5.959.1 Detailed Description

```
template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename _Rx_traits = regex_traits<_Ch_type>>
class std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >
```

Iterates over submatches in a range (or *splits* a text string).

The purpose of this iterator is to enumerate all, or all specified, matches of a regular expression within a text range. The dereferenced value of an iterator of this class is a `std::sub_match` object.

Definition at line 2562 of file `regex.h`.

5.959.2 Constructor & Destructor Documentation

```
5.959.2.1 template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename _Rx_traits
= regex_traits<_Ch_type>> std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >::regex_token_iterator
( ) [inline]
```

Default constructs a `regex_token_iterator`.

A default-constructed `regex_token_iterator` is a singular iterator that will compare equal to the one-past-the-end value for any iterator of the same type.

Definition at line 2580 of file `regex.h`.

```
5.959.2.2 template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename _Rx_traits
= regex_traits<_Ch_type>>> std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >::regex_token_iterator
( _Bi_iter __a, _Bi_iter __b, const regex_type & __re, int __submatch = 0, regex_constants::match_flag_type
__m = regex_constants::match_default ) [inline]
```

Constructs a regex_token_iterator...

Parameters

__a	[IN] The start of the text to search.
__b	[IN] One-past-the-end of the text to search.
__re	[IN] The regular expression to search for.
__submatch	[IN] Which submatch to return. There are some special values for this parameter: <ul style="list-style-type: none"> • -1 each enumerated subexpression does NOT match the regular expression (aka field splitting) • 0 the entire string matching the subexpression is returned for each match within the text. • >0 enumerates only the indicated subexpression from a match within the text.
__m	[IN] Policy flags for match rules.

Definition at line 2602 of file regex.h.

```
5.959.2.3 template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename _Rx_traits
= regex_traits<_Ch_type>>> std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >::regex_token_iterator
( _Bi_iter __a, _Bi_iter __b, const regex_type & __re, const std::vector< int > & __submatches,
regex_constants::match_flag_type __m = regex_constants::match_default ) [inline]
```

Constructs a regex_token_iterator...

Parameters

__a	[IN] The start of the text to search.
__b	[IN] One-past-the-end of the text to search.
__re	[IN] The regular expression to search for.
__submatches	[IN] A list of subexpressions to return for each regular expression match within the text.
__m	[IN] Policy flags for match rules.

Definition at line 2618 of file regex.h.

```
5.959.2.4 template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename
_Rx_traits = regex_traits<_Ch_type>>> std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits
>::regex_token_iterator ( _Bi_iter __a, _Bi_iter __b, const regex_type & __re, initializer_list< int >
__submatches, regex_constants::match_flag_type __m = regex_constants::match_default ) [inline]
```

Constructs a regex_token_iterator...

Parameters

<code>__a</code>	[IN] The start of the text to search.
<code>__b</code>	[IN] One-past-the-end of the text to search.
<code>__re</code>	[IN] The regular expression to search for.
<code>__submatches</code>	[IN] A list of subexpressions to return for each regular expression match within the text.
<code>__m</code>	[IN] Policy flags for match rules.

Definition at line 2635 of file `regex.h`.

```
5.959.2.5  template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename _Rx_traits
           = regex_traits<_Ch_type>> template<std::size_t _Nm> std::regex_token_iterator<_Bi_iter, _Ch_type, _Rx_traits
           >::regex_token_iterator ( _Bi_iter __a, _Bi_iter __b, const regex_type & __re, const int(&) __submatches[ _Nm],
           regex_constants::match_flag_type __m = regex_constants::match_default )  [inline]
```

Constructs a `regex_token_iterator`...

Parameters

<code>__a</code>	[IN] The start of the text to search.
<code>__b</code>	[IN] One-past-the-end of the text to search.
<code>__re</code>	[IN] The regular expression to search for.
<code>__submatches</code>	[IN] A list of subexpressions to return for each regular expression match within the text.
<code>__m</code>	[IN] Policy flags for match rules.

Definition at line 2653 of file `regex.h`.

References `std::regex_constants::match_default`.

```
5.959.2.6  template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename _Rx_traits
           = regex_traits<_Ch_type>> std::regex_token_iterator<_Bi_iter, _Ch_type, _Rx_traits>::regex_token_iterator
           ( const regex_token_iterator<_Bi_iter, _Ch_type, _Rx_traits> & __rhs )  [inline]
```

Copy constructs a `regex_token_iterator`.

Parameters

<code>__rhs</code>	[IN] A <code>regex_token_iterator</code> to copy.
--------------------	---

Definition at line 2685 of file `regex.h`.

5.959.3 Member Function Documentation

```
5.959.3.1 template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename _Rx_traits
= regex_traits<_Ch_type>> bool std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator!=( const
regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits > & __rhs ) const [inline]
```

Compares a regex_token_iterator to another for inequality.

Definition at line 2707 of file regex.h.

```
5.959.3.2 template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename _Rx_traits
= regex_traits<_Ch_type>> const value_type& std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits
>::operator*( ) const [inline]
```

Dereferences a regex_token_iterator.

Definition at line 2714 of file regex.h.

```
5.959.3.3 template<typename _Bi_iter, typename _Ch_type, typename _Rx_traits > regex_token_iterator< _Bi_iter, _Ch_type,
_Rx_traits > & std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator++( )
```

Increments a regex_token_iterator.

Definition at line 607 of file regex.tcc.

Referenced by std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator==().

```
5.959.3.4 template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename _Rx_traits
= regex_traits<_Ch_type>> regex_token_iterator std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits
>::operator++( int ) [inline]
```

Postincrements a regex_token_iterator.

Definition at line 2734 of file regex.h.

```
5.959.3.5 template<typename _Bi_iter, typename _Ch_type = typename iterator_traits<_Bi_iter>::value_type, typename _Rx_traits
= regex_traits<_Ch_type>> const value_type* std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits
>::operator->( ) const [inline]
```

Selects a regex_token_iterator member.

Definition at line 2721 of file regex.h.

```
5.959.3.6 template<typename _Bi_iter, typename _Ch_type, typename _Rx_traits > regex_token_iterator< _Bi_iter,
_Ch_type, _Rx_traits > & std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >::operator=( const
regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits > & __rhs )
```

Assigns a regex_token_iterator to another.

Parameters

<code>__rhs</code>	[IN] A <code>regex_token_iterator</code> to copy.
--------------------	---

Definition at line 571 of file `regex.tcc`.

References `std::regex_token_iterator<_Bi_iter, _Ch_type, _Rx_traits>::operator==()`.

Referenced by `std::regex_iterator<_Bi_iter, _Ch_type, _Rx_traits>::operator++()`.

5.959.3.7 `template<typename _Bi_iter, typename _Ch_type, typename _Rx_traits> bool std::regex_token_iterator<_Bi_iter, _Ch_type, _Rx_traits>::operator==(const regex_token_iterator<_Bi_iter, _Ch_type, _Rx_traits> & __rhs) const`

Compares a `regex_token_iterator` to another for equality.

Definition at line 587 of file `regex.tcc`.

References `std::regex_token_iterator<_Bi_iter, _Ch_type, _Rx_traits>::operator++()`.

Referenced by `std::regex_token_iterator<_Bi_iter, _Ch_type, _Rx_traits>::operator=()`.

The documentation for this class was generated from the following files:

- [regex.h](#)
- [regex.tcc](#)

5.960 `std::regex_traits<_Ch_type>` Struct Template Reference

Public Types

- typedef `_RegexMask` **char_class_type**
- typedef `_Ch_type` **char_type**
- typedef [std::locale](#) **locale_type**
- typedef [std::basic_string](#)< `char_type` > **string_type**

Public Member Functions

- [regex_traits](#) ()
- [locale_type](#) `getloc` () const
- [locale_type](#) `imbue` ([locale_type](#) __loc)
- bool `isctype` (`_Ch_type` __c, `char_class_type` __f) const
- template<typename `_Fwd_iter` >
`char_class_type` `lookup_classname` (`_Fwd_iter` __first, `_Fwd_iter` __last, bool __icase=false) const
- template<typename `_Fwd_iter` >
[string_type](#) `lookup_collatename` (`_Fwd_iter` __first, `_Fwd_iter` __last) const
- template<typename `_Fwd_iter` >
[string_type](#) `transform` (`_Fwd_iter` __first, `_Fwd_iter` __last) const
- template<typename `_Fwd_iter` >
[string_type](#) `transform_primary` (`_Fwd_iter` __first, `_Fwd_iter` __last) const
- `char_type` `translate` (`char_type` __c) const
- `char_type` `translate_nocase` (`char_type` __c) const
- int `value` (`_Ch_type` __ch, int __radix) const

Static Public Member Functions

- static std::size_t [length](#) (const char_type *__p)

Protected Attributes

- [locale_type](#) [_M_locale](#)

5.960.1 Detailed Description

```
template<typename _Ch_type>
struct std::regex_traits<_Ch_type>
```

Describes aspects of a regular expression.

A regular expression traits class that satisfies the requirements of section [28.7].

The class regex is parameterized around a set of related types and functions used to complete the definition of its semantics. This class satisfies the requirements of such a traits class.

Definition at line 87 of file regex.h.

5.960.2 Constructor & Destructor Documentation

```
5.960.2.1 template<typename _Ch_type> std::regex_traits<_Ch_type>::regex_traits ( ) [inline]
```

Constructs a default traits object.

Definition at line 164 of file regex.h.

5.960.3 Member Function Documentation

```
5.960.3.1 template<typename _Ch_type> locale_type std::regex_traits<_Ch_type>::getloc ( ) const [inline]
```

Gets a copy of the current locale in use by the regex_traits object.

Definition at line 377 of file regex.h.

References `std::regex_constants::awk`, `std::regex_constants::basic`, `std::regex_constants::collate`, `std::regex_constants::ECMAScript`, `std::regex_constants::egrep`, `std::regex_constants::extended`, `std::regex_constants::grep`, `std::regex_constants::icase`, `std::regex_constants::nosubs`, and `std::regex_constants::optimize`.

Referenced by `std::regex_traits<_Ch_type>::value()`.

```
5.960.3.2 template<typename _Ch_type> locale_type std::regex_traits<_Ch_type>::imbue ( locale_type __loc )
[inline]
```

Imbues the regex_traits object with a copy of a new locale.

Parameters

<code>__loc</code>	A locale.
--------------------	-----------

Returns

a copy of the previous locale in use by the `regex_traits` object.

Note

Calling `imbue` with a different locale than the one currently in use invalidates all cached data held by `*this`.

Definition at line 366 of file `regex.h`.

5.960.3.3 `template<typename _Ch_type> bool std::regex_traits<_Ch_type>::isctype (_Ch_type __c, char_class_type __f) const`

Determines if `c` is a member of an identified class.

Parameters

<code>__c</code>	a character.
<code>__f</code>	a class type (as returned from <code>lookup_classname</code>).

Returns

true if the character `__c` is a member of the classification represented by `__f`, false otherwise.

Exceptions

<code>std::bad_cast</code>	if the current locale does not have a ctype facet.
----------------------------	--

Definition at line 331 of file `regex.tcc`.

References `std::regex_traits<_Ch_type>::value()`.

Referenced by `std::regex_traits<_Ch_type>::lookup_classname()`.

5.960.3.4 `template<typename _Ch_type> static std::size_t std::regex_traits<_Ch_type>::length (const char_type * __p) [inline], [static]`

Gives the length of a C-style string starting at `__p`.

Parameters

<code>__p</code>	a pointer to the start of a character sequence.
------------------	---

Returns

the number of characters between `*__p` and the first default-initialized value of type `char_type`. In other words, uses the C-string algorithm for determining the length of a sequence of characters.

Definition at line 177 of file `regex.h`.

```
5.960.3.5 template<typename _Ch_type > template<typename _Fwd_iter > regex_traits<_Ch_type>::char_class_type
std::regex_traits<_Ch_type>::lookup_classname ( _Fwd_iter __first, _Fwd_iter __last, bool __icase = false )
const
```

Maps one or more characters to a named character classification.

Parameters

<code>__first</code>	beginning of the character sequence.
<code>__last</code>	one-past-the-end of the character sequence.
<code>__icase</code>	ignores the case of the classification name.

Returns

an unspecified value that represents the character classification named by the character sequence designated by the iterator range `[__first, __last)`. If `icase` is true, the returned mask identifies the classification regardless of the case of the characters to be matched (for example, `[[:lower:]]` is the same as `[[:alpha:]]`), otherwise a case-dependent classification is returned. The value returned shall be independent of the case of the characters in the character sequence. If the name is not recognized then returns a value that compares equal to 0.

At least the following names (or their wide-character equivalent) are supported.

- `d`
- `w`
- `s`
- `alnum`
- `alpha`
- `blank`
- `cntrl`
- `digit`
- `graph`

- lower
- print
- punct
- space
- upper
- xdigit

Definition at line 287 of file regex.tcc.

References `std::regex_traits<_Ch_type>::isctype()`.

Referenced by `std::regex_traits<_Ch_type>::lookup_collatename()`.

5.960.3.6 `template<typename _Ch_type> template<typename _Fwd_iter> regex_traits<_Ch_type>::string_type
std::regex_traits<_Ch_type>::lookup_collatename (_Fwd_iter __first, _Fwd_iter __last) const`

Gets a collation element by name.

Parameters

<code>__first</code>	beginning of the collation element name.
<code>__last</code>	one-past-the-end of the collation element name.

Returns

a sequence of one or more characters that represents the collating element consisting of the character sequence designated by the iterator range [`__first`, `__last`). Returns an empty string if the character sequence is not a valid collating element.

Definition at line 131 of file regex.tcc.

References `std::regex_traits<_Ch_type>::lookup_classname()`.

5.960.3.7 `template<typename _Ch_type> template<typename _Fwd_iter> string_type std::regex_traits<_Ch_type>
>::transform (_Fwd_iter __first, _Fwd_iter __last) const [inline]`

Gets a sort key for a character sequence.

Parameters

<code>__first</code>	beginning of the character sequence.
<code>__last</code>	one-past-the-end of the character sequence.

Returns a sort key for the character sequence designated by the iterator range [`F1`, `F2`) such that if the character sequence [`G1`, `G2`) sorts before the character sequence [`H1`, `H2`) then `v.transform(G1, G2) < v.transform(H1, H2)`.

What this really does is provide a more efficient way to compare a string to multiple other strings in locales with fancy collation rules and equivalence classes.

Returns

a locale-specific sort key equivalent to the input range.

Exceptions

<code>std::bad_cast</code>	if the current locale does not have a collate facet.
----------------------------	--

Definition at line 230 of file `regex.h`.

References `std::basic_string<_CharT, _Traits, _Alloc>::data()`, and `std::basic_string<_CharT, _Traits, _Alloc>::size()`.

5.960.3.8 `template<typename _Ch_type> template<typename _Fwd_iter> string_type std::regex_traits<_Ch_type>::transform_primary (_Fwd_iter __first, _Fwd_iter __last) const [inline]`

Gets a sort key for a character sequence, independent of case.

Parameters

<code>__first</code>	beginning of the character sequence.
<code>__last</code>	one-past-the-end of the character sequence.

Effects: if `typeid(use_facet<collate<_Ch_type>>) == typeid(collate_byname<_Ch_type>)` and the form of the sort key returned by `collate_byname<_Ch_type>::transform(__first, __last)` is known and can be converted into a primary sort key then returns that key, otherwise returns an empty string.

Todo Implement this function correctly.

Definition at line 254 of file `regex.h`.

References `std::vector<_Tp, _Alloc>::data()`, and `std::vector<_Tp, _Alloc>::size()`.

5.960.3.9 `template<typename _Ch_type> char_type std::regex_traits<_Ch_type>::translate (char_type __c) const [inline]`

Performs the identity translation.

Parameters

<code>__c</code>	A character to the locale-specific character set.
------------------	---

Returns

__c.

Definition at line 188 of file regex.h.

```
5.960.3.10  template<typename _Ch_type> char_type std::regex_traits<_Ch_type>::translate_nocase ( char_type __c )
            const [inline]
```

Translates a character into a case-insensitive equivalent.

Parameters

__c	A character to the locale-specific character set.
-----	---

Returns

the locale-specific lower-case equivalent of __c.

Exceptions

<i>std::bad_cast</i>	if the imbued locale does not support the ctype facet.
----------------------	--

Definition at line 201 of file regex.h.

```
5.960.3.11  template<typename _Ch_type> int std::regex_traits<_Ch_type>::value ( _Ch_type __ch, int __radix ) const
```

Converts a digit to an int.

Parameters

__ch	a character representing a digit.
__radix	the radix if the numeric conversion (limited to 8, 10, or 16).

Returns

the value represented by the digit __ch in base radix if the character __ch is a valid digit in base radix; otherwise returns -1.

Definition at line 345 of file regex.tcc.

References `std::basic_ios<_CharT, _Traits>::fail()`, `std::match_results<_Bi_iter, _Alloc>::format()`, `std::regex_constants::format_sed`, `std::regex_traits<_Ch_type>::getloc()`, `std::hex()`, `std::oct()`, and `std::regex_traits<_Ch_type>::value()`.

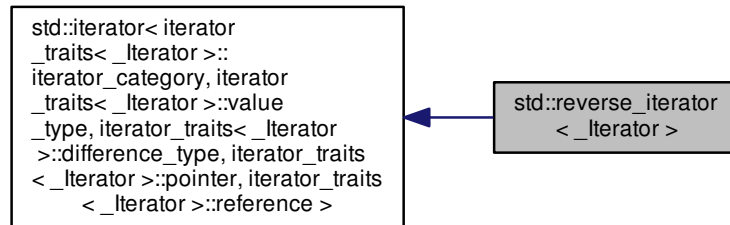
Referenced by `std::regex_traits<_Ch_type>::isctype()`, and `std::regex_traits<_Ch_type>::value()`.

The documentation for this struct was generated from the following files:

- [regex.h](#)
- [regex.tcc](#)

5.961 `std::reverse_iterator<_Iterator>` Class Template Reference

Inheritance diagram for `std::reverse_iterator<_Iterator>`:



Public Types

- typedef `__traits_type::difference_type` **difference_type**
- typedef `iterator_traits<_Iterator>::iterator_category` **iterator_category**
- typedef `_Iterator` **iterator_type**
- typedef `__traits_type::pointer` **pointer**
- typedef `__traits_type::reference` **reference**
- typedef `iterator_traits<_Iterator>::value_type` **value_type**

Public Member Functions

- [reverse_iterator](#) ()
- [reverse_iterator](#) (iterator_type __x)
- [reverse_iterator](#) (const [reverse_iterator](#) &__x)
- template<typename _Iter >
 [reverse_iterator](#) (const [reverse_iterator](#)<_Iter> &__x)
- iterator_type [base](#) () const
- reference [operator*](#) () const
- [reverse_iterator](#) [operator++](#) (difference_type __n) const
- [reverse_iterator](#) & [operator++](#) ()
- [reverse_iterator](#) [operator++](#) (int)
- [reverse_iterator](#) & [operator+=](#) (difference_type __n)
- [reverse_iterator](#) [operator--](#) (difference_type __n) const
- [reverse_iterator](#) & [operator--](#) ()
- [reverse_iterator](#) [operator--](#) (int)
- [reverse_iterator](#) & [operator-=](#) (difference_type __n)
- pointer [operator->](#) () const
- reference [operator\[\]](#) (difference_type __n) const

Protected Types

- `typedef iterator_traits< _Iterator > __traits_type`

Protected Attributes

- `_Iterator current`

5.961.1 Detailed Description

```
template<typename _Iterator>
class std::reverse_iterator< _Iterator >
```

Bidirectional and random access iterators have corresponding reverse iterator adaptors that iterate through the data structure in the opposite direction. They have the same signatures as the corresponding iterators. The fundamental relation between a reverse iterator and its corresponding iterator `i` is established by the identity:

```
&*(reverse_iterator(i)) == &(i - 1)
```

This mapping is dictated by the fact that while there is always a pointer past the end of an array, there might not be a valid pointer before the beginning of an array. [24.4.1]/1,2

Reverse iterators can be tricky and surprising at first. Their semantics make sense, however, and the trickiness is a side effect of the requirement that the iterators must be safe.

Definition at line 97 of file `bits/stl_iterator.h`.

5.961.2 Member Typedef Documentation

5.961.2.1 `typedef iterator_traits< _Iterator >::iterator_category std::iterator< iterator_traits< _Iterator >::iterator_category, iterator_traits< _Iterator >::value_type, iterator_traits< _Iterator >::difference_type, iterator_traits< _Iterator >::pointer, iterator_traits< _Iterator >::reference>::iterator_category` [inherited]

One of the [tag types](#).

Definition at line 121 of file `stl_iterator_base_types.h`.

5.961.2.2 `typedef iterator_traits< _Iterator >::value_type std::iterator< iterator_traits< _Iterator >::iterator_category, iterator_traits< _Iterator >::value_type, iterator_traits< _Iterator >::difference_type, iterator_traits< _Iterator >::pointer, iterator_traits< _Iterator >::reference>::value_type` [inherited]

The type "pointed to" by the iterator.

Definition at line 123 of file `stl_iterator_base_types.h`.

5.961.3 Constructor & Destructor Documentation

5.961.3.1 `template<typename _Iterator> std::reverse_iterator<_Iterator>::reverse_iterator () [inline]`

The default constructor value-initializes member `current`. If it is a pointer, that means it is zero-initialized.

Definition at line 121 of file `bits/stl_iterator.h`.

Referenced by `std::reverse_iterator<_Iterator>::operator+()`, and `std::reverse_iterator<_Iterator>::operator-()`.

5.961.3.2 `template<typename _Iterator> std::reverse_iterator<_Iterator>::reverse_iterator (iterator_type __x) [inline], [explicit]`

This iterator will move in the opposite direction that `x` does.

Definition at line 127 of file `bits/stl_iterator.h`.

5.961.3.3 `template<typename _Iterator> std::reverse_iterator<_Iterator>::reverse_iterator (const reverse_iterator<_Iterator> &__x) [inline]`

The copy constructor is normal.

Definition at line 132 of file `bits/stl_iterator.h`.

5.961.3.4 `template<typename _Iterator> template<typename _Iter> std::reverse_iterator<_Iterator>::reverse_iterator (const reverse_iterator<_Iter> &__x) [inline]`

A `reverse_iterator` across other types can be copied if the underlying iterator can be converted to the type of `current`.

Definition at line 140 of file `bits/stl_iterator.h`.

5.961.4 Member Function Documentation

5.961.4.1 `template<typename _Iterator> iterator_type std::reverse_iterator<_Iterator>::base () const [inline]`

Returns

`current`, the iterator used for underlying work.

Definition at line 147 of file `bits/stl_iterator.h`.

Referenced by `std::inserter()`, `std::make_reverse_iterator()`, and `std::operator==()`.

5.961.4.2 `template<typename _Iterator> reference std::reverse_iterator<_Iterator>::operator*() const [inline]`

Returns

A reference to the value at `-current`

This requires that `-current` is dereferenceable.

Warning

This implementation requires that for an iterator of the underlying iterator type, `x`, a reference obtained by `*x` remains valid after `x` has been modified or destroyed. This is a bug: <http://gcc.gnu.org/PR51823>

Definition at line 161 of file `bits/stl_iterator.h`.

Referenced by `std::inserter()`, and `std::reverse_iterator<_Iterator>::operator->()`.

5.961.4.3 `template<typename _Iterator> reverse_iterator std::reverse_iterator<_Iterator>::operator+(difference_type __n) const [inline]`

Returns

A `reverse_iterator` that refers to `current - __n`

The underlying iterator must be a Random Access Iterator.

Definition at line 232 of file `bits/stl_iterator.h`.

References `std::reverse_iterator<_Iterator>::reverse_iterator()`.

Referenced by `std::inserter()`, and `std::operator==()`.

5.961.4.4 `template<typename _Iterator> reverse_iterator& std::reverse_iterator<_Iterator>::operator++() [inline]`

Returns

`*this`

Decrements the underlying iterator.

Definition at line 182 of file `bits/stl_iterator.h`.

Referenced by `std::inserter()`.

5.961.4.5 `template<typename _Iterator> reverse_iterator std::reverse_iterator<_Iterator>::operator++(int) [inline]`

Returns

The original value of `*this`

Decrements the underlying iterator.

Definition at line 194 of file `bits/stl_iterator.h`.

5.961.4.6 `template<typename _Iterator> reverse_iterator& std::reverse_iterator<_Iterator>::operator+=(difference_type __n) [inline]`

Returns

`*this`

Moves the underlying iterator backwards `__n` steps. The underlying iterator must be a Random Access Iterator.

Definition at line 242 of file `bits/stl_iterator.h`.

Referenced by `std::inserter()`.

5.961.4.7 `template<typename _Iterator> reverse_iterator std::reverse_iterator<_Iterator>::operator- (difference_type __n) const [inline]`

Returns

A `reverse_iterator` that refers to `current - __n`

The underlying iterator must be a Random Access Iterator.

Definition at line 254 of file `bits/stl_iterator.h`.

References `std::reverse_iterator<_Iterator>::reverse_iterator()`.

Referenced by `std::inserter()`, and `std::operator==()`.

5.961.4.8 `template<typename _Iterator> reverse_iterator& std::reverse_iterator<_Iterator>::operator-- () [inline]`

Returns

`*this`

Increments the underlying iterator.

Definition at line 207 of file `bits/stl_iterator.h`.

Referenced by `std::inserter()`.

5.961.4.9 `template<typename _Iterator> reverse_iterator std::reverse_iterator<_Iterator>::operator-- (int) [inline]`

Returns

A `reverse_iterator` with the previous value of `*this`

Increments the underlying iterator.

Definition at line 219 of file `bits/stl_iterator.h`.

5.961.4.10 `template<typename _Iterator> reverse_iterator& std::reverse_iterator<_Iterator>::operator-= (difference_type __n) [inline]`

Returns

`*this`

Moves the underlying iterator forwards `__n` steps. The underlying iterator must be a Random Access Iterator.

Definition at line 264 of file `bits/stl_iterator.h`.

Referenced by `std::inserter()`.

5.961.4.11 `template<typename _Iterator> pointer std::reverse_iterator<_Iterator>::operator-> () const [inline]`

Returns

A pointer to the value at `-current`

This requires that `-current` is dereferenceable.

Definition at line 173 of file `bits/stl_iterator.h`.

References `std::reverse_iterator<_Iterator>::operator*()`.

Referenced by `std::inserter()`.

5.961.4.12 `template<typename _Iterator> reference std::reverse_iterator<_Iterator>::operator[] (difference_type __n) const [inline]`

Returns

The value at `current - __n - 1`

The underlying iterator must be a Random Access Iterator.

Definition at line 276 of file `bits/stl_iterator.h`.

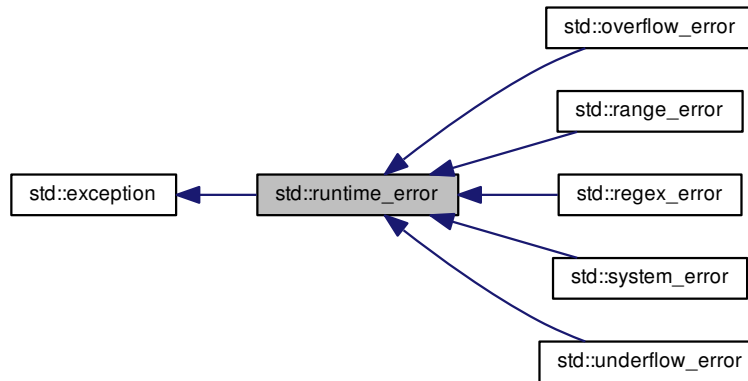
Referenced by `std::inserter()`.

The documentation for this class was generated from the following file:

- [bits/stl_iterator.h](#)

5.962 std::runtime_error Class Reference

Inheritance diagram for std::runtime_error:



Public Member Functions

- `runtime_error` (const [string](#) &__arg) _GLIBCXX_TXN_SAFE
- `runtime_error` (const char *) _GLIBCXX_TXN_SAFE
- virtual const char * `what` () const _GLIBCXX_TXN_SAFE_DYN noexcept

5.962.1 Detailed Description

One of two subclasses of exception.

Runtime errors represent problems outside the scope of a program; they cannot be easily predicted and can generally only be caught as the program executes.

Definition at line 197 of file `stdexcept`.

5.962.2 Constructor & Destructor Documentation

5.962.2.1 std::runtime_error::runtime_error (const string &__arg) [explicit]

Takes a character string describing the error.

5.962.3 Member Function Documentation

5.962.3.1 `virtual const char* std::runtime_error::what () const` `[virtual],[noexcept]`

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

The documentation for this class was generated from the following file:

- [stdexcept](#)

5.963 `std::scoped_allocator_adaptor< _OuterAlloc, _InnerAllocs >` Class Template Reference

Inherits `_OuterAlloc`.

Public Types

- `typedef __traits::const_pointer const_pointer`
- `typedef __traits::const_void_pointer const_void_pointer`
- `typedef __traits::difference_type difference_type`
- `typedef __inner_type::__type inner_allocator_type`
- `typedef __and< typename __traits::is_always_equal, typename allocator_traits< _InnerAllocs >::is_always_equal... >::type is_always_equal`
- `typedef _OuterAlloc outer_allocator_type`
- `typedef __traits::pointer pointer`
- `typedef __or< typename __traits::propagate_on_container_copy_assignment, typename allocator_traits< _InnerAllocs >::propagate_on_container_copy_assignment... >::type propagate_on_container_copy_assignment`
- `typedef __or< typename __traits::propagate_on_container_move_assignment, typename allocator_traits< _InnerAllocs >::propagate_on_container_move_assignment... >::type propagate_on_container_move_assignment`
- `typedef __or< typename __traits::propagate_on_container_swap, typename allocator_traits< _InnerAllocs >::propagate_on_container_swap... >::type propagate_on_container_swap`
- `typedef __traits::size_type size_type`
- `typedef __traits::value_type value_type`
- `typedef __traits::void_pointer void_pointer`

Public Member Functions

- template<typename _Outer2, typename = _Constructible<_Outer2>>>
scoped_allocator_adaptor (_Outer2 && __outer, const _InnerAllocs &... __inner)
- **scoped_allocator_adaptor** (const [scoped_allocator_adaptor](#) & __other)
- **scoped_allocator_adaptor** ([scoped_allocator_adaptor](#) && __other)
- template<typename _Outer2, typename = _Constructible<const _Outer2&>>>
scoped_allocator_adaptor (const [scoped_allocator_adaptor](#)<_Outer2, _InnerAllocs... > & __other)
- template<typename _Outer2, typename = _Constructible<_Outer2>>>
scoped_allocator_adaptor ([scoped_allocator_adaptor](#)<_Outer2, _InnerAllocs... > && __other)
- pointer **allocate** (size_type __n)
- pointer **allocate** (size_type __n, const_void_pointer __hint)
- template<typename _Tp, typename... _Args>
void **construct** (_Tp * __p, _Args &&... __args)
- template<typename _T1, typename _T2, typename... _Args1, typename... _Args2>
void **construct** ([pair](#)<_T1, _T2 > * __p, [piecewise_construct_t](#), [tuple](#)<_Args1... > __x, [tuple](#)<_Args2... > __y)
- template<typename _T1, typename _T2 >
void **construct** ([pair](#)<_T1, _T2 > * __p)
- template<typename _T1, typename _T2, typename _Up, typename _Vp >
void **construct** ([pair](#)<_T1, _T2 > * __p, _Up && __u, _Vp && __v)
- template<typename _T1, typename _T2, typename _Up, typename _Vp >
void **construct** ([pair](#)<_T1, _T2 > * __p, const [pair](#)<_Up, _Vp > & __x)
- template<typename _T1, typename _T2, typename _Up, typename _Vp >
void **construct** ([pair](#)<_T1, _T2 > * __p, [pair](#)<_Up, _Vp > && __x)
- void **deallocate** (pointer __p, size_type __n)
- template<typename _Tp >
void **destroy** (_Tp * __p)
- inner_allocator_type & **inner_allocator** () noexcept
- const inner_allocator_type & **inner_allocator** () const noexcept
- size_type **max_size** () const
- [scoped_allocator_adaptor](#) & **operator=** (const [scoped_allocator_adaptor](#) &)=default
- [scoped_allocator_adaptor](#) & **operator=** ([scoped_allocator_adaptor](#) &&)=default
- outer_allocator_type & **outer_allocator** () noexcept
- const outer_allocator_type & **outer_allocator** () const noexcept
- [scoped_allocator_adaptor](#) **select_on_container_copy_construction** () const

Friends

- template<typename _Outer, typename... _Inner>
class **scoped_allocator_adaptor**
- template<typename... >
class **__inner_type_impl**
- template<typename _OutA1, typename _OutA2, typename... _InA>
bool **operator==** (const [scoped_allocator_adaptor](#)<_OutA1, _InA... > & __a, const [scoped_allocator_adaptor](#)<_OutA2, _InA... > & __b) noexcept

5.963.1 Detailed Description

```
template<typename _OuterAlloc, typename... _InnerAllocs>
class std::scoped_allocator_adaptor< _OuterAlloc, _InnerAllocs >
```

Primary class template.

Definition at line 83 of file `scoped_allocator`.

The documentation for this class was generated from the following file:

- [scoped_allocator](#)

5.964 std::seed_seq Class Reference

Public Types

- typedef uint_least32_t [result_type](#)

Public Member Functions

- [seed_seq](#) () noexcept
- template<typename _IntType >
 [seed_seq](#) (std::initializer_list< _IntType > il)
- template<typename _InputIterator >
 [seed_seq](#) (_InputIterator __begin, _InputIterator __end)
- [seed_seq](#) (const [seed_seq](#) &)=delete
- template<typename _RandomAccessIterator >
 void **generate** (_RandomAccessIterator __begin, _RandomAccessIterator __end)
- [seed_seq](#) & **operator=** (const [seed_seq](#) &)=delete
- template<typename OutputIterator >
 void **param** (OutputIterator __dest) const
- size_t **size** () const noexcept

5.964.1 Detailed Description

The `seed_seq` class generates sequences of seeds for random number generators.

Definition at line 5860 of file `random.h`.

5.964.2 Member Typedef Documentation

5.964.2.1 typedef uint_least32_t std::seed_seq::result_type

The type of the seed vales.

Definition at line 5864 of file `random.h`.

5.964.3 Constructor & Destructor Documentation

5.964.3.1 `std::seed_seq::seed_seq()` `[inline]`, `[noexcept]`

Default constructor.

Definition at line 5867 of file `random.h`.

Referenced by `std::operator>>()`.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.965 `std::set<_Key, _Compare, _Alloc>` Class Template Reference

Public Types

- typedef `_Key` [key_type](#)
- typedef `_Key` [value_type](#)
- typedef `_Compare` [key_compare](#)
- typedef `_Compare` [value_compare](#)
- typedef `_Alloc` [allocator_type](#)

- typedef `_Alloc_traits::pointer` [pointer](#)
- typedef `_Alloc_traits::const_pointer` [const_pointer](#)
- typedef `_Alloc_traits::reference` [reference](#)
- typedef `_Alloc_traits::const_reference` [const_reference](#)
- typedef `_Rep_type::const_iterator` [iterator](#)
- typedef `_Rep_type::const_iterator` [const_iterator](#)
- typedef `_Rep_type::const_reverse_iterator` [reverse_iterator](#)
- typedef `_Rep_type::const_reverse_iterator` [const_reverse_iterator](#)
- typedef `_Rep_type::size_type` [size_type](#)
- typedef `_Rep_type::difference_type` [difference_type](#)

Public Member Functions

- [set](#) () noexcept(*/*conditional */*)
- [set](#) (const [_Compare](#) &__comp, const [allocator_type](#) &__a=[allocator_type](#)())
- [template](#)<typename [_InputIterator](#) >
 [set](#) ([_InputIterator](#) __first, [_InputIterator](#) __last)
- [template](#)<typename [_InputIterator](#) >
 [set](#) ([_InputIterator](#) __first, [_InputIterator](#) __last, const [_Compare](#) &__comp, const [allocator_type](#) &__a=[allocator_type](#)↵
 [_type](#)())
- [set](#) (const [set](#) &__x)
- [set](#) ([set](#) &&__x) noexcept(is_nothrow_copy_constructible< [_Compare](#) >::value)
- [set](#) ([initializer_list](#)< [value_type](#) > __l, const [_Compare](#) &__comp=[_Compare](#)(), const [allocator_type](#) &__a↵
 a=[allocator_type](#)())
- [set](#) (const [allocator_type](#) &__a)
- [set](#) (const [set](#) &__x, const [allocator_type](#) &__a)
- [set](#) ([set](#) &&__x, const [allocator_type](#) &__a) noexcept(is_nothrow_copy_constructible< [_Compare](#) >::value &&↵
 [_Alloc_traits](#)::S_always_equal())
- [set](#) ([initializer_list](#)< [value_type](#) > __l, const [allocator_type](#) &__a)
- [template](#)<typename [_InputIterator](#) >
 [set](#) ([_InputIterator](#) __first, [_InputIterator](#) __last, const [allocator_type](#) &__a)
- [iterator begin](#) () const noexcept
- [iterator cbegin](#) () const noexcept
- [iterator cend](#) () const noexcept
- [void clear](#) () noexcept
- [reverse_iterator crbegin](#) () const noexcept
- [reverse_iterator crend](#) () const noexcept
- [template](#)<typename... [_Args](#)>
 std::pair< [iterator](#), bool > [emplace](#) ([_Args](#) &&...__args)
- [template](#)<typename... [_Args](#)>
 [iterator emplace_hint](#) (const [iterator](#) __pos, [_Args](#) &&...__args)
- [bool empty](#) () const noexcept
- [iterator end](#) () const noexcept
- [_GLIBCXX_ABI_TAG_CXX11 iterator erase](#) (const [iterator](#) __position)
- [size_type erase](#) (const [key_type](#) &__x)
- [_GLIBCXX_ABI_TAG_CXX11 iterator erase](#) (const [iterator](#) __first, const [iterator](#) __last)
- [allocator_type get_allocator](#) () const noexcept
- std::pair< [iterator](#), bool > [insert](#) (const [value_type](#) &__x)
- std::pair< [iterator](#), bool > [insert](#) ([value_type](#) &&__x)
- [iterator insert](#) (const [iterator](#) __position, const [value_type](#) &__x)
- [iterator insert](#) (const [iterator](#) __position, [value_type](#) &&__x)
- [template](#)<typename [_InputIterator](#) >
 void [insert](#) ([_InputIterator](#) __first, [_InputIterator](#) __last)
- [void insert](#) ([initializer_list](#)< [value_type](#) > __l)
- [key_compare key_comp](#) () const
- [size_type max_size](#) () const noexcept
- [set & operator=](#) (const [set](#) &__x)
- [set & operator=](#) ([set](#) &&)=default
- [set & operator=](#) ([initializer_list](#)< [value_type](#) > __l)
- [reverse_iterator rbegin](#) () const noexcept
- [reverse_iterator rend](#) () const noexcept
- [size_type size](#) () const noexcept

- void `swap` (`set &__x`) noexcept(*/*conditional */*)
- `value_compare` `value_comp` () const
- `size_type` `count` (const `key_type` &__x) const
- template<typename _Kt >
auto `count` (const _Kt &__x) const -> decltype(_M_t._M_count_tr(__x))
- `iterator` `find` (const `key_type` &__x)
- `const_iterator` `find` (const `key_type` &__x) const
- template<typename _Kt >
auto `find` (const _Kt &__x) -> decltype(`iterator`
- template<typename _Kt >
auto `find` (const _Kt &__x) const -> decltype(`const_iterator`
- `iterator` `lower_bound` (const `key_type` &__x)
- `const_iterator` `lower_bound` (const `key_type` &__x) const
- template<typename _Kt >
auto `lower_bound` (const _Kt &__x) -> decltype(_M_t._M_lower_bound_tr(__x))
- template<typename _Kt >
auto `lower_bound` (const _Kt &__x) const -> decltype(_M_t._M_lower_bound_tr(__x))
- `iterator` `upper_bound` (const `key_type` &__x)
- `const_iterator` `upper_bound` (const `key_type` &__x) const
- template<typename _Kt >
auto `upper_bound` (const _Kt &__x) -> decltype(_M_t._M_upper_bound_tr(__x))
- template<typename _Kt >
auto `upper_bound` (const _Kt &__x) const -> decltype(_M_t._M_upper_bound_tr(__x))
- `std::pair`< `iterator`, `iterator` > `equal_range` (const `key_type` &__x)
- `std::pair`< `const_iterator`, `const_iterator` > `equal_range` (const `key_type` &__x) const
- template<typename _Kt >
auto `equal_range` (const _Kt &__x) -> decltype(_M_t._M_equal_range_tr(__x))
- template<typename _Kt >
auto `equal_range` (const _Kt &__x) const -> decltype(_M_t._M_equal_range_tr(__x))

Friends

- template<typename _K1, typename _C1, typename _A1 >
bool **operator**< (const `set`< _K1, _C1, _A1 > &, const `set`< _K1, _C1, _A1 > &)
- template<typename _K1, typename _C1, typename _A1 >
bool **operator**== (const `set`< _K1, _C1, _A1 > &, const `set`< _K1, _C1, _A1 > &)

5.965.1 Detailed Description

```
template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>>
class std::set< _Key, _Compare, _Alloc >
```

A standard container made up of unique keys, which can be retrieved in logarithmic time.

Template Parameters

<code>_Key</code>	Type of key objects.
<code>_Compare</code>	Comparison function object type, defaults to <code>less<_Key></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_Key></code> .

Meets the requirements of a [container](#), a [reversible container](#), and an [associative container](#) (using unique keys).

Sets support bidirectional iterators.

The private tree data is declared exactly the same way for set and multiset; the distinction is made entirely in how the tree functions are called (`*_unique` versus `*_equal`, same as the standard).

Definition at line 90 of file `stl_set.h`.

5.965.2 Member Typedef Documentation

5.965.2.1 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> typedef _Alloc std::set<_Key, _Compare, _Alloc>::allocator_type`

Public typedefs.

Definition at line 107 of file `stl_set.h`.

5.965.2.2 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> typedef _Rep_type::const_iterator std::set<_Key, _Compare, _Alloc>::const_iterator`

Iterator-related typedefs.

Definition at line 131 of file `stl_set.h`.

5.965.2.3 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> typedef _Alloc_traits::const_pointer std::set<_Key, _Compare, _Alloc>::const_pointer`

Iterator-related typedefs.

Definition at line 124 of file `stl_set.h`.

5.965.2.4 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> typedef _Alloc_traits::const_reference std::set<_Key, _Compare, _Alloc>::const_reference`

Iterator-related typedefs.

Definition at line 126 of file `stl_set.h`.

5.965.2.5 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> typedef
_Rep_type::const_reverse_iterator std::set< _Key, _Compare, _Alloc >::const_reverse_iterator`

Iterator-related typedefs.

Definition at line 133 of file stl_set.h.

5.965.2.6 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> typedef
_Rep_type::difference_type std::set< _Key, _Compare, _Alloc >::difference_type`

Iterator-related typedefs.

Definition at line 135 of file stl_set.h.

5.965.2.7 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> typedef
_Rep_type::const_iterator std::set< _Key, _Compare, _Alloc >::iterator`

Iterator-related typedefs.

Definition at line 130 of file stl_set.h.

5.965.2.8 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> typedef
_Compare std::set< _Key, _Compare, _Alloc >::key_compare`

Public typedefs.

Definition at line 105 of file stl_set.h.

5.965.2.9 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> typedef
_Key std::set< _Key, _Compare, _Alloc >::key_type`

Public typedefs.

Definition at line 103 of file stl_set.h.

5.965.2.10 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> typedef
_Alloc_traits::pointer std::set< _Key, _Compare, _Alloc >::pointer`

Iterator-related typedefs.

Definition at line 123 of file stl_set.h.

5.965.2.11 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> typedef
_Alloc_traits::reference std::set< _Key, _Compare, _Alloc >::reference`

Iterator-related typedefs.

Definition at line 125 of file stl_set.h.

5.965.2.12 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> typedef
_Rep_type::const_reverse_iterator std::set<_Key, _Compare, _Alloc>::reverse_iterator`

Iterator-related typedefs.

Definition at line 132 of file `stl_set.h`.

5.965.2.13 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> typedef
_Rep_type::size_type std::set<_Key, _Compare, _Alloc>::size_type`

Iterator-related typedefs.

Definition at line 134 of file `stl_set.h`.

5.965.2.14 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> typedef
_Compare std::set<_Key, _Compare, _Alloc>::value_compare`

Public typedefs.

Definition at line 106 of file `stl_set.h`.

5.965.2.15 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> typedef
_Key std::set<_Key, _Compare, _Alloc>::value_type`

Public typedefs.

Definition at line 104 of file `stl_set.h`.

5.965.3 Constructor & Destructor Documentation

5.965.3.1 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::set<_Key, _Compare, _Alloc>::set () [inline], [noexcept]`

Default constructor creates no elements.

Definition at line 142 of file `stl_set.h`.

5.965.3.2 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::set<_Key, _Compare, _Alloc>::set (const _Compare & __comp, const allocator_type & __a =
allocator_type()) [inline], [explicit]`

Creates a set with no elements.

Parameters

<code>__comp</code>	Comparator to use.
<code>__a</code>	An allocator object.

Definition at line 154 of file stl_set.h.

```
5.965.3.3 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename _InputIterator > std::set<_Key, _Compare, _Alloc>::set ( _InputIterator __first, _InputIterator
__last ) [inline]
```

Builds a set from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Create a set consisting of copies of the elements from [`__first`,`__last`). This is linear in N if the range is already sorted, and NlogN otherwise (where N is distance(`__first`,`__last`)).

Definition at line 169 of file stl_set.h.

```
5.965.3.4 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename _InputIterator > std::set<_Key, _Compare, _Alloc>::set ( _InputIterator __first, _InputIterator
__last, const _Compare & __comp, const allocator_type & __a = allocator_type() ) [inline]
```

Builds a set from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__comp</code>	A comparison functor.
<code>__a</code>	An allocator object.

Create a set consisting of copies of the elements from [`__first`,`__last`). This is linear in N if the range is already sorted, and NlogN otherwise (where N is distance(`__first`,`__last`)).

Definition at line 186 of file stl_set.h.

```
5.965.3.5 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::set<_Key, _Compare, _Alloc>::set ( const set<_Key, _Compare, _Alloc> & __x ) [inline]
```

Set copy constructor.

Parameters

<code>__x</code>	A set of identical element and allocator types.
------------------	---

The newly-created set uses a copy of the allocation object used by `__x`.

Definition at line 199 of file stl_set.h.

5.965.3.6 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::set< _Key, _Compare, _Alloc >::set (set< _Key, _Compare, _Alloc > && __x) [inline], [noexcept]`

Set move constructor

Parameters

<code>__x</code>	A set of identical element and allocator types.
------------------	---

The newly-created set contains the exact contents of x. The contents of x are a valid, but unspecified set.

Definition at line 210 of file stl_set.h.

5.965.3.7 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::set< _Key, _Compare, _Alloc >::set (initializer_list< value_type > __l, const _Compare & __comp =
_Compare(), const allocator_type & __a = allocator_type()) [inline]`

Builds a set from an initializer_list.

Parameters

<code>__l</code>	An initializer_list.
<code>__comp</code>	A comparison functor.
<code>__a</code>	An allocator object.

Create a set consisting of copies of the elements in the list. This is linear in N if the list is already sorted, and NlogN otherwise (where N is `__l.size()`).

Definition at line 224 of file stl_set.h.

5.965.3.8 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::set< _Key, _Compare, _Alloc >::set (const allocator_type & __a) [inline], [explicit]`

Allocator-extended default constructor.

Definition at line 232 of file stl_set.h.

5.965.3.9 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
std::set< _Key, _Compare, _Alloc >::set (const set< _Key, _Compare, _Alloc > & __x, const allocator_type & __a
) [inline]`

Allocator-extended copy constructor.

Definition at line 236 of file stl_set.h.

```
5.965.3.10 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
    std::set<_Key, _Compare, _Alloc>::set ( set<_Key, _Compare, _Alloc> && __x, const allocator_type & __a )
    [inline], [noexcept]
```

Allocator-extended move constructor.

Definition at line 240 of file stl_set.h.

```
5.965.3.11 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
    std::set<_Key, _Compare, _Alloc>::set ( initializer_list<value_type> __l, const allocator_type & __a )
    [inline]
```

Allocator-extended initializer-list constructor.

Definition at line 246 of file stl_set.h.

```
5.965.3.12 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
    template<typename _InputIterator> std::set<_Key, _Compare, _Alloc>::set ( _InputIterator __first, _InputIterator
    __last, const allocator_type & __a ) [inline]
```

Allocator-extended range constructor.

Definition at line 252 of file stl_set.h.

5.965.4 Member Function Documentation

```
5.965.4.1 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> iterator
    std::set<_Key, _Compare, _Alloc>::begin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the set. Iteration is done in ascending order according to the keys.

Definition at line 317 of file stl_set.h.

```
5.965.4.2 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> iterator
    std::set<_Key, _Compare, _Alloc>::cbegin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the set. Iteration is done in ascending order according to the keys.

Definition at line 354 of file stl_set.h.

```
5.965.4.3 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> iterator
    std::set<_Key, _Compare, _Alloc>::cend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the set. Iteration is done in ascending order according to the keys.

Definition at line 363 of file stl_set.h.

5.965.4.4 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> void
std::set<_Key, _Compare, _Alloc>::clear () [inline], [noexcept]`

Erases all elements in a set. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 650 of file `stl_set.h`.

5.965.4.5 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
size_type std::set<_Key, _Compare, _Alloc>::count (const key_type & __x) const [inline]`

Finds the number of elements.

Parameters

\leftrightarrow	Element to located.
x	

Returns

Number of elements with specified key.

This function only makes sense for multisets; for set the result will either be 0 (not present) or 1 (present).

Definition at line 665 of file stl_set.h.

```
5.965.4.6 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>>
template<typename _Kt > auto std::set<_Key, _Compare, _Alloc >::count ( const _Kt & __x ) const ->
decltype(_M_t._M_count_tr(__x)) [inline]
```

Finds the number of elements.

Parameters

\leftrightarrow	Element to located.
x	

Returns

Number of elements with specified key.

This function only makes sense for multisets; for set the result will either be 0 (not present) or 1 (present).

Definition at line 671 of file stl_set.h.

```
5.965.4.7 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>>
reverse_iterator std::set<_Key, _Compare, _Alloc >::crbegin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the last element in the set. Iteration is done in descending order according to the keys.

Definition at line 372 of file stl_set.h.

```
5.965.4.8 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>>
reverse_iterator std::set<_Key, _Compare, _Alloc >::crend ( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last pair in the set. Iteration is done in descending order according to the keys.

Definition at line 381 of file stl_set.h.

```
5.965.4.9 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>>
template<typename... _Args> std::pair<iterator, bool> std::set<_Key, _Compare, _Alloc >::emplace ( _Args
&&... __args ) [inline]
```

Attempts to build and insert an element into the set.

Parameters

<code>__args</code>	Arguments used to generate an element.
---------------------	--

Returns

A pair, of which the first element is an iterator that points to the possibly inserted element, and the second is a bool that is true if the element was actually inserted.

This function attempts to build and insert an element into the set. A set relies on unique keys and thus an element is only inserted if it is not already present in the set.

Insertion requires logarithmic time.

Definition at line 433 of file `stl_set.h`.

```
5.965.4.10 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            template<typename... _Args> iterator std::set<_Key, _Compare, _Alloc>::emplace_hint( const_iterator __pos,
            _Args &&... __args ) [inline]
```

Attempts to insert an element into the set.

Parameters

<code>__pos</code>	An iterator that serves as a hint as to where the element should be inserted.
<code>__args</code>	Arguments used to generate the element to be inserted.

Returns

An iterator that points to the element with key equivalent to the one generated from `__args` (may or may not be the element itself).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `emplace()` does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints

Insertion requires logarithmic time (if the hint is not taken).

Definition at line 459 of file `stl_set.h`.

```
5.965.4.11 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> bool
            std::set<_Key, _Compare, _Alloc>::empty( ) const [inline], [noexcept]
```

Returns true if the set is empty.

Definition at line 387 of file `stl_set.h`.

5.965.4.12 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
 iterator std::set<_Key, _Compare, _Alloc>::end () const [inline], [noexcept]`

Returns a read-only (constant) iterator that points one past the last element in the set. Iteration is done in ascending order according to the keys.

Definition at line 326 of file stl_set.h.

5.965.4.13 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
 std::pair<iterator, iterator> std::set<_Key, _Compare, _Alloc>::equal_range (const key_type & __x)
 [inline]`

Finds a subsequence matching given key.

Parameters

<code>__x</code>	Key to be located.
------------------	--------------------

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
               c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multisets.

Definition at line 796 of file stl_set.h.

5.965.4.14 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
 std::pair<const_iterator, const_iterator> std::set<_Key, _Compare, _Alloc>::equal_range (const
 key_type & __x) const [inline]`

Finds a subsequence matching given key.

Parameters

<code>__x</code>	Key to be located.
------------------	--------------------

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
              c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multisets.

Definition at line 800 of file stl_set.h.

```
5.965.4.15  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            template<typename _Kt > auto std::set<_Key, _Compare, _Alloc>::equal_range ( const _Kt & __x ) ->
            decltype(_M_t._M_equal_range_tr(__x))  [inline]
```

Finds a subsequence matching given key.

Parameters

<code>__x</code>	Key to be located.
------------------	--------------------

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
              c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multisets.

Definition at line 806 of file stl_set.h.

```
5.965.4.16  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            template<typename _Kt > auto std::set<_Key, _Compare, _Alloc>::equal_range ( const _Kt & __x ) const ->
            decltype(_M_t._M_equal_range_tr(__x))  [inline]
```

Finds a subsequence matching given key.

Parameters

<code>_↔</code>	Key to be located.
<code>_x</code>	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function is equivalent to

```
std::make_pair(c.lower_bound(val),
              c.upper_bound(val))
```

(but is faster than making the calls separately).

This function probably only makes sense for multisets.

Definition at line 812 of file stl_set.h.

```
5.965.4.17 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
             _GLIBCXX_ABI_TAG_CXX11 iterator std::set< _Key, _Compare, _Alloc >::erase ( const_iterator __position )
             [inline]
```

Erases an element from a set.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following `__position` prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from a set. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 571 of file stl_set.h.

Referenced by `std::set< _Key, _Compare, _Alloc >::erase()`.

```
5.965.4.18 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
             size_type std::set< _Key, _Compare, _Alloc >::erase ( const key_type & __x ) [inline]
```

Erases elements according to the provided key.

Parameters

<code>_↔</code>	Key of element to be erased.
<code>_x</code>	

Returns

The number of elements erased.

This function erases all the elements located by the given key from a set. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 601 of file `stl_set.h`.

```
5.965.4.19  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            _GLIBCXX_ABI_TAG_CXX11 iterator std::set< _Key, _Compare, _Alloc >::erase ( const_iterator __first,
            const_iterator __last ) [inline]
```

Erases a [`__first`,`__last`) range of elements from a set.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be erased.
<code>__last</code>	Iterator pointing to the end of the range to be erased.

Returns

The iterator `__last`.

This function erases a sequence of elements from a set. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 623 of file `stl_set.h`.

References `std::set< _Key, _Compare, _Alloc >::erase()`.

```
5.965.4.20  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            iterator std::set< _Key, _Compare, _Alloc >::find ( const key_type & __x ) [inline]
```

Tries to locate an element in a set.

Parameters

<code>_↔</code>	Element to be located.
<code>_x</code>	

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (end()) iterator.

Definition at line 692 of file stl_set.h.

```
5.965.4.21  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            const_iterator std::set<_Key, _Compare, _Alloc>::find ( const key_type & __x ) const    [inline]
```

Tries to locate an element in a set.

Parameters

$_x$	Element to be located.
-------	------------------------

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (end()) iterator.

Definition at line 696 of file stl_set.h.

```
5.965.4.22  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            template<typename _Kt > auto std::set<_Key, _Compare, _Alloc>::find ( const _Kt & __x )-> decltype(iterator
            [inline]
```

Tries to locate an element in a set.

Parameters

$_x$	Element to be located.
-------	------------------------

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (end()) iterator.

Definition at line 702 of file stl_set.h.

```
5.965.4.23  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            template<typename _Kt > auto std::set< _Key, _Compare, _Alloc >::find ( const _Kt & __x ) const ->
            decltype(const_iterator [inline]
```

Tries to locate an element in a set.

Parameters

\leftarrow	Element to be located.
$_x$	

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (end()) iterator.

Definition at line 708 of file stl_set.h.

```
5.965.4.24  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            allocator_type std::set< _Key, _Compare, _Alloc >::get_allocator ( ) const [inline], [noexcept]
```

Returns the allocator object with which the set was constructed.

Definition at line 308 of file stl_set.h.

```
5.965.4.25  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            std::pair<iterator, bool> std::set< _Key, _Compare, _Alloc >::insert ( const value_type & __x ) [inline]
```

Attempts to insert an element into the set.

Parameters

\leftarrow	Element to be inserted.
$_x$	

Returns

A pair, of which the first element is an iterator that points to the possibly inserted element, and the second is a bool that is true if the element was actually inserted.

This function attempts to insert an element into the set. A set relies on unique keys and thus an element is only inserted if it is not already present in the set.

Insertion requires logarithmic time.

Definition at line 480 of file stl_set.h.

References std::pair< _T1, _T2 >::first, and std::pair< _T1, _T2 >::second.

Referenced by std::set< _Key, _Compare, _Alloc >::insert().

```
5.965.4.26 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            iterator std::set< _Key, _Compare, _Alloc >::insert ( const_iterator __position, const value_type & __x )
            [inline]
```

Attempts to insert an element into the set.

Parameters

<code>__position</code>	An iterator that serves as a hint as to where the element should be inserted.
<code>__x</code>	Element to be inserted.

Returns

An iterator that points to the element with key of `__x` (may or may not be the element passed in).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `insert()` does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints

Insertion requires logarithmic time (if the hint is not taken).

Definition at line 517 of file `stl_set.h`.

References `std::set< _Key, _Compare, _Alloc >::insert()`.

```
5.965.4.27 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            template<typename _InputIterator > void std::set< _Key, _Compare, _Alloc >::insert ( _InputIterator __first,
            _InputIterator __last ) [inline]
```

A template function that attempts to insert a range of elements.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be inserted.
<code>__last</code>	Iterator pointing to the end of the range.

Complexity similar to that of the range constructor.

Definition at line 537 of file `stl_set.h`.

```
5.965.4.28 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> void
            std::set< _Key, _Compare, _Alloc >::insert ( initializer_list< value_type > __l ) [inline]
```

Attempts to insert a list of elements into the set.

Parameters

↩	A std::initializer_list<value_type> of elements to be inserted.
↩	
↩	
↩	
/	

Complexity similar to that of the range constructor.

Definition at line 549 of file stl_set.h.

References std::set<_Key, _Compare, _Alloc >::insert().

5.965.4.29 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
key_compare std::set<_Key, _Compare, _Alloc >::key_comp () const [inline]`

Returns the comparison object with which the set was constructed.

Definition at line 300 of file stl_set.h.

5.965.4.30 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
iterator std::set<_Key, _Compare, _Alloc >::lower_bound (const key_type &__x) [inline]`

Finds the beginning of a subsequence matching given key.

Parameters

↩	Key to be located.
__x	

Returns

Iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

Definition at line 727 of file stl_set.h.

5.965.4.31 `template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
const_iterator std::set<_Key, _Compare, _Alloc >::lower_bound (const key_type &__x) const [inline]`

Finds the beginning of a subsequence matching given key.

Parameters

$_x$	Key to be located.
-------	--------------------

Returns

Iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

Definition at line 731 of file stl_set.h.

```
5.965.4.32  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            template<typename _Kt > auto std::set<_Key, _Compare, _Alloc >::lower_bound ( const _Kt & __x ) ->
            decltype(_M_t._M_lower_bound_tr(__x)) [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

$_x$	Key to be located.
-------	--------------------

Returns

Iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

Definition at line 737 of file stl_set.h.

```
5.965.4.33  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            template<typename _Kt > auto std::set<_Key, _Compare, _Alloc >::lower_bound ( const _Kt & __x ) const ->
            decltype(_M_t._M_lower_bound_tr(__x)) [inline]
```

Finds the beginning of a subsequence matching given key.

Parameters

$_x$	Key to be located.
-------	--------------------

Returns

Iterator pointing to first element equal to or greater than key, or end().

This function returns the first element of a subsequence of elements that matches the given key. If unsuccessful it returns an iterator pointing to the first element that has a greater value than given key or end() if no such element exists.

Definition at line 743 of file stl_set.h.

```
5.965.4.34  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            size_type std::set<_Key, _Compare, _Alloc>::max_size( ) const    [inline], [noexcept]
```


Returns the maximum size of the set.

Definition at line 397 of file stl_set.h.

```
5.965.4.35  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> set&
            std::set<_Key, _Compare, _Alloc>::operator=( const set<_Key, _Compare, _Alloc> &__x )    [inline]
```

Set assignment operator.

Parameters

	A set of identical element and allocator types.
<code>__x</code>	

All the elements of `__x` are copied, but unlike the copy constructor, the allocator object is not copied.

Definition at line 266 of file stl_set.h.



```
5.965.4.36  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> set&
            std::set<_Key, _Compare, _Alloc>::operator=( set<_Key, _Compare, _Alloc> && )    [default]
```

Move assignment operator.

```
5.965.4.37  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>> set&
            std::set<_Key, _Compare, _Alloc>::operator=( initializer_list<value_type> __l )    [inline]
```

Set list assignment operator.

Parameters

	An initializer_list.
<code>__l</code>	
	
<code>__l</code>	
<code>l</code>	

This function fills a set with copies of the elements in the initializer list `__l`.

Note that the assignment completely changes the set and that the resulting set's size is the same as the number of elements assigned. Old data may be lost.

Definition at line 289 of file `stl_set.h`.

```
5.965.4.38 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
reverse_iterator std::set<_Key, _Compare, _Alloc>::rbegin( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the last element in the set. Iteration is done in descending order according to the keys.

Definition at line 335 of file `stl_set.h`.

```
5.965.4.39 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
reverse_iterator std::set<_Key, _Compare, _Alloc>::rend( ) const [inline], [noexcept]
```

Returns a read-only (constant) reverse iterator that points to the last pair in the set. Iteration is done in descending order according to the keys.

Definition at line 344 of file `stl_set.h`.

```
5.965.4.40 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
size_type std::set<_Key, _Compare, _Alloc>::size( ) const [inline], [noexcept]
```

Returns the size of the set.

Definition at line 392 of file `stl_set.h`.

```
5.965.4.41 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
void std::set<_Key, _Compare, _Alloc>::swap( set<_Key, _Compare, _Alloc> & __x ) [inline],
[noexcept]
```

Swaps data with another set.

Parameters

<code>__x</code>	A set of the same element and allocator types.
------------------	--

This exchanges the elements between two sets in constant time. (It is only swapping a pointer, an integer, and an instance of the `Compare` type (which itself is often stateless and empty), so it should be quite fast.) Note that the global `std::swap()` function is specialized such that `std::swap(s1,s2)` will feed to this function.

Definition at line 412 of file `stl_set.h`.

```
5.965.4.42 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
iterator std::set<_Key, _Compare, _Alloc>::upper_bound( const key_type & __x ) [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>_↔</code>	Key to be located.
<code>__x</code>	

Returns

Iterator pointing to the first element greater than key, or end().

Definition at line 757 of file stl_set.h.

```
5.965.4.43  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            const_iterator std::set<_Key, _Compare, _Alloc>::upper_bound ( const key_type & __x ) const  [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>_↔</code>	Key to be located.
<code>__x</code>	

Returns

Iterator pointing to the first element greater than key, or end().

Definition at line 761 of file stl_set.h.

```
5.965.4.44  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            template<typename _Kt > auto std::set<_Key, _Compare, _Alloc>::upper_bound ( const _Kt & __x ) ->
            decltype(_M_t._M_upper_bound_tr(__x))  [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>_↔</code>	Key to be located.
<code>__x</code>	

Returns

Iterator pointing to the first element greater than key, or end().

Definition at line 767 of file stl_set.h.

```
5.965.4.45 template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
template<typename _Kt > auto std::set<_Key, _Compare, _Alloc>::upper_bound ( const _Kt & __x ) const ->
decltype(_M_t._M_upper_bound_tr(__x)) [inline]
```

Finds the end of a subsequence matching given key.

Parameters

<code>_↔</code>	Key to be located.
<code>_X</code>	

Returns

Iterator pointing to the first element greater than key, or end().

Definition at line 773 of file `stl_set.h`.

```
5.965.4.46  template<typename _Key, typename _Compare = std::less<_Key>, typename _Alloc = std::allocator<_Key>>
            value_compare std::set<_Key, _Compare, _Alloc>::value_comp ( ) const  [inline]
```

Returns the comparison object with which the set was constructed.

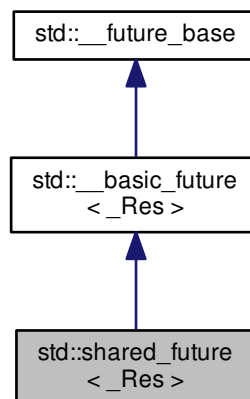
Definition at line 304 of file `stl_set.h`.

The documentation for this class was generated from the following file:

- [stl_set.h](#)

5.966 std::shared_future<_Res> Class Template Reference

Inheritance diagram for `std::shared_future<_Res>`:



Public Types

- template<typename _Res >
using **_Ptr** = [unique_ptr](#)< _Res, _Result_base::_Deleter >
- using **_State_base** = _State_baseV2

Public Member Functions

- [shared_future](#) (const [shared_future](#) &__sf)
- [shared_future](#) ([future](#)< _Res > &&__uf) noexcept
- [shared_future](#) ([shared_future](#) &&__sf) noexcept
- const _Res & [get](#) () const
- [shared_future](#) & **operator=** (const [shared_future](#) &__sf)
- [shared_future](#) & **operator=** ([shared_future](#) &&__sf) noexcept
- bool **valid** () const noexcept
- void **wait** () const
- template<typename _Rep , typename _Period >
[future_status](#) **wait_for** (const [chrono::duration](#)< _Rep, _Period > &__rel) const
- template<typename _Clock , typename _Duration >
[future_status](#) **wait_until** (const [chrono::time_point](#)< _Clock, _Duration > &__abs) const

Static Public Member Functions

- template<typename _Res , typename _Allocator >
static **_Ptr**< [_Result_alloc](#)< _Res, _Allocator > > **_S_allocate_result** (const _Allocator &__a)
- template<typename _Res , typename _Tp >
static **_Ptr**< [_Result](#)< _Res > > **_S_allocate_result** (const [std::allocator](#)< _Tp > &__a)
- template<typename _BoundFn >
static [std::shared_ptr](#)< _State_base > **_S_make_async_state** (_BoundFn &&__fn)
- template<typename _BoundFn >
static [std::shared_ptr](#)< _State_base > **_S_make_deferred_state** (_BoundFn &&__fn)
- template<typename _Res_ptr , typename _BoundFn >
static **_Task_setter**< _Res_ptr, _BoundFn > **_S_task_setter** (_Res_ptr &__ptr, _BoundFn &__call)

Protected Types

- typedef [_future_base::_Result](#)< _Res > & **__result_type**
- typedef [shared_ptr](#)< _State_base > **__state_type**

Protected Member Functions

- [__result_type](#) **_M_get_result** () const
- void **_M_swap** ([__basic_future](#) &__that) noexcept

5.966.1 Detailed Description

```
template<typename _Res>
class std::shared_future< _Res >
```

Primary template for shared_future.

Definition at line 118 of file future.

5.966.2 Member Typedef Documentation

5.966.2.1 `template<typename _Res > using std::__future_base::Ptr = unique_ptr<_Res, _Result_base::Deleter>`
[inherited]

A unique_ptr for result objects.

Definition at line 214 of file future.

5.966.3 Constructor & Destructor Documentation

5.966.3.1 `template<typename _Res > std::shared_future< _Res >::shared_future (const shared_future< _Res > & __sf)` [inline]

Copy constructor.

Definition at line 869 of file future.

5.966.3.2 `template<typename _Res > std::shared_future< _Res >::shared_future (future< _Res > && __uf)`
[inline], [noexcept]

Construct from a future rvalue.

Definition at line 872 of file future.

5.966.3.3 `template<typename _Res > std::shared_future< _Res >::shared_future (shared_future< _Res > && __sf)`
[inline], [noexcept]

Construct from a shared_future rvalue.

Definition at line 877 of file future.

5.966.4 Member Function Documentation

5.966.4.1 `template<typename _Res> __result_type std::__basic_future< _Res >::M_get_result () const` [inline],
[protected], [inherited]

Wait for the state to be ready and rethrow any stored exception.

Definition at line 684 of file future.

5.966.4.2 template<typename _Res > const _Res& std::shared_future<_Res >::get () const [inline]

Retrieving the value.

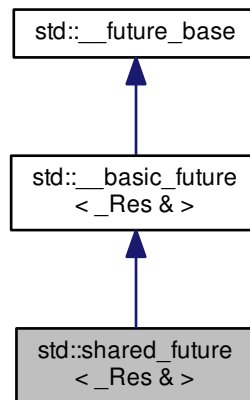
Definition at line 895 of file future.

The documentation for this class was generated from the following file:

- [future](#)

5.967 std::shared_future<_Res & > Class Template Reference

Inheritance diagram for std::shared_future<_Res & >:



Public Types

- template<typename _Res >
using `_Ptr` = `unique_ptr<_Res, _Result_base::Deleter >`
- using `_State_base` = `_State_baseV2`

Public Member Functions

- `shared_future` (const `shared_future` & __sf)
- `shared_future` (`future<_Res & >` && __uf) noexcept
- `shared_future` (`shared_future` && __sf) noexcept
- `_Res` & `get` () const
- `shared_future` & `operator=` (const `shared_future` & __sf)
- `shared_future` & `operator=` (`shared_future` && __sf) noexcept
- bool `valid` () const noexcept
- void `wait` () const
- `future_status` `wait_for` (const `chrono::duration<_Rep, _Period >` & __rel) const
- `future_status` `wait_until` (const `chrono::time_point<_Clock, _Duration >` & __abs) const

Static Public Member Functions

- `template<typename _Res, typename _Allocator >`
`static _Ptr< _Result_alloc< _Res, _Allocator > > _S_allocate_result (const _Allocator &__a)`
- `template<typename _Res, typename _Tp >`
`static _Ptr< _Result< _Res > > _S_allocate_result (const std::allocator< _Tp > &__a)`
- `template<typename _BoundFn >`
`static std::shared_ptr< _State_base > _S_make_async_state (_BoundFn &&__fn)`
- `template<typename _BoundFn >`
`static std::shared_ptr< _State_base > _S_make_deferred_state (_BoundFn &&__fn)`
- `template<typename _Res_ptr, typename _BoundFn >`
`static _Task_setter< _Res_ptr, _BoundFn > _S_task_setter (_Res_ptr &__ptr, _BoundFn &__call)`

Protected Types

- `typedef __future_base:: Result< _Res & > & __result_type`
- `typedef shared_ptr< _State_base > __state_type`

Protected Member Functions

- `__result_type _M_get_result () const`
- `void _M_swap (__basic_future &__that) noexcept`

5.967.1 Detailed Description

```
template<typename _Res>
class std::shared_future< _Res & >
```

Partial specialization for `shared_future<R&>`

Definition at line 900 of file `future`.

5.967.2 Member Typedef Documentation

5.967.2.1 `template<typename _Res > using std::__future_base::Ptr = unique_ptr<_Res, _Result_base:: Deleter>`
`[inherited]`

A `unique_ptr` for result objects.

Definition at line 214 of file `future`.

5.967.3 Constructor & Destructor Documentation

5.967.3.1 `template<typename _Res > std::shared_future< _Res & >::shared_future (const shared_future< _Res & >`
`&__sf) [inline]`

Copy constructor.

Definition at line 908 of file `future`.

5.967.3.2 `template<typename _Res> std::shared_future< _Res >::shared_future (future< _Res > && __uf)`
`[inline], [noexcept]`

Construct from a future rvalue.

Definition at line 911 of file future.

5.967.3.3 `template<typename _Res> std::shared_future< _Res >::shared_future (shared_future< _Res > && __sf)` `[inline], [noexcept]`

Construct from a shared_future rvalue.

Definition at line 916 of file future.

5.967.4 Member Function Documentation

5.967.4.1 `__result_type std::__basic_future< _Res >::M_get_result () const` `[inline], [protected], [inherited]`

Wait for the state to be ready and rethrow any stored exception.

Definition at line 684 of file future.

5.967.4.2 `template<typename _Res> _Res& std::shared_future< _Res >::get () const` `[inline]`

Retrieving the value.

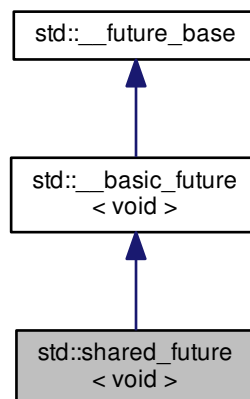
Definition at line 934 of file future.

The documentation for this class was generated from the following file:

- [future](#)

5.968 std::shared_future< void > Class Template Reference

Inheritance diagram for std::shared_future< void >:



Public Types

- `template<typename _Res >`
`using _Ptr = unique_ptr< _Res, _Result_base::_Deleter >`
- `using _State_base = _State_baseV2`

Public Member Functions

- `shared_future (const shared_future &__sf)`
- `shared_future (future< void > &&__uf) noexcept`
- `shared_future (shared_future &&__sf) noexcept`
- `void get () const`
- `shared_future & operator= (const shared_future &__sf)`
- `shared_future & operator= (shared_future &&__sf) noexcept`
- `bool valid () const noexcept`
- `void wait () const`
- `future_status wait_for (const chrono::duration< _Rep, _Period > &__rel) const`
- `future_status wait_until (const chrono::time_point< _Clock, _Duration > &__abs) const`

Static Public Member Functions

- `template<typename _Res, typename _Allocator >`
`static _Ptr< _Result_alloc< _Res, _Allocator > > _S_allocate_result (const _Allocator &__a)`
- `template<typename _Res, typename _Tp >`
`static _Ptr< _Result< _Res > > _S_allocate_result (const std::allocator< _Tp > &__a)`
- `template<typename _BoundFn >`
`static std::shared_ptr< _State_base > _S_make_async_state (_BoundFn &&__fn)`
- `template<typename _BoundFn >`
`static std::shared_ptr< _State_base > _S_make_deferred_state (_BoundFn &&__fn)`
- `template<typename _Res_ptr, typename _BoundFn >`
`static _Task_setter< _Res_ptr, _BoundFn > _S_task_setter (_Res_ptr &__ptr, _BoundFn &__call)`

Protected Types

- `typedef _future_base::_Result< void > & __result_type`
- `typedef shared_ptr< _State_base > __state_type`

Protected Member Functions

- `__result_type _M_get_result () const`
- `void _M_swap (__basic_future &__that) noexcept`

5.968.1 Detailed Description

```
template<>
class std::shared_future< void >
```

Explicit specialization for `shared_future<void>`

Definition at line 939 of file future.

5.968.2 Member Typedef Documentation

5.968.2.1 `template<typename _Res > using std::__future_base::_Ptr = unique_ptr<_Res, _Result_base::_Deleter>`
[[inherited](#)]

A `unique_ptr` for result objects.

Definition at line 214 of file `future`.

5.968.3 Constructor & Destructor Documentation

5.968.3.1 `std::shared_future< void >::shared_future (const shared_future< void > &__sf)` [[inline](#)]

Copy constructor.

Definition at line 947 of file `future`.

5.968.3.2 `std::shared_future< void >::shared_future (future< void > &&__uf)` [[inline](#)], [[noexcept](#)]

Construct from a future rvalue.

Definition at line 950 of file `future`.

5.968.3.3 `std::shared_future< void >::shared_future (shared_future< void > &&__sf)` [[inline](#)],
[[noexcept](#)]

Construct from a `shared_future` rvalue.

Definition at line 955 of file `future`.

5.968.4 Member Function Documentation

5.968.4.1 `__result_type std::__basic_future< void >::M_get_result () const` [[inline](#)], [[protected](#)],
[[inherited](#)]

Wait for the state to be ready and rethrow any stored exception.

Definition at line 684 of file `future`.

The documentation for this class was generated from the following file:

- [future](#)

5.969 std::shared_lock< _Mutex > Class Template Reference

Public Types

- typedef `_Mutex` **mutex_type**

Public Member Functions

- **shared_lock** (mutex_type &__m)
- **shared_lock** (mutex_type &__m, [defer_lock_t](#)) noexcept
- **shared_lock** (mutex_type &__m, [try_to_lock_t](#))
- **shared_lock** (mutex_type &__m, [adopt_lock_t](#))
- template<typename _Clock , typename _Duration >
shared_lock (mutex_type &__m, const [chrono::time_point](#)< _Clock, _Duration > &__abs_time)
- template<typename _Rep , typename _Period >
shared_lock (mutex_type &__m, const [chrono::duration](#)< _Rep, _Period > &__rel_time)
- **shared_lock** ([shared_lock](#) const &)=delete
- **shared_lock** ([shared_lock](#) &&__sl) noexcept
- void **lock** ()
- mutex_type * **mutex** () const noexcept
- **operator bool** () const noexcept
- [shared_lock](#) & **operator=** ([shared_lock](#) const &)=delete
- [shared_lock](#) & **operator=** ([shared_lock](#) &&__sl) noexcept
- bool **owns_lock** () const noexcept
- mutex_type * **release** () noexcept
- void **swap** ([shared_lock](#) &__u) noexcept
- bool **try_lock** ()
- template<typename _Rep , typename _Period >
bool **try_lock_for** (const [chrono::duration](#)< _Rep, _Period > &__rel_time)
- template<typename _Clock , typename _Duration >
bool **try_lock_until** (const [chrono::time_point](#)< _Clock, _Duration > &__abs_time)
- void **unlock** ()

5.969.1 Detailed Description

```
template<typename _Mutex>
class std::shared_lock< _Mutex >
```

`shared_lock`

Definition at line 543 of file `shared_mutex`.

The documentation for this class was generated from the following file:

- [shared_mutex](#)

5.970 std::shared_ptr< _Tp > Class Template Reference

Inherits std::__shared_ptr< _Tp, _Lp >.

Public Types

- typedef `_Tp element_type`

Public Member Functions

- constexpr `shared_ptr()` noexcept
- `shared_ptr` (const `shared_ptr` &) noexcept=default
- template<typename `_Tp1` >
`shared_ptr` (`_Tp1 *__p`)
- template<typename `_Tp1`, typename `_Deleter` >
`shared_ptr` (`_Tp1 *__p`, `_Deleter __d`)
- template<typename `_Deleter` >
`shared_ptr` (nullptr_t `__p`, `_Deleter __d`)
- template<typename `_Tp1`, typename `_Deleter`, typename `_Alloc` >
`shared_ptr` (`_Tp1 *__p`, `_Deleter __d`, `_Alloc __a`)
- template<typename `_Deleter`, typename `_Alloc` >
`shared_ptr` (nullptr_t `__p`, `_Deleter __d`, `_Alloc __a`)
- template<typename `_Tp1` >
`shared_ptr` (const `shared_ptr`< `_Tp1` > &`__r`, `_Tp *__p`) noexcept
- template<typename `_Tp1`, typename = `_Convertible<_Tp1*>>`
`shared_ptr` (const `shared_ptr`< `_Tp1` > &`__r`) noexcept
- `shared_ptr` (`shared_ptr` &&`__r`) noexcept
- template<typename `_Tp1`, typename = `_Convertible<_Tp1*>>`
`shared_ptr` (`shared_ptr`< `_Tp1` > &&`__r`) noexcept
- template<typename `_Tp1` >
`shared_ptr` (const `weak_ptr`< `_Tp1` > &`__r`)
- template<typename `_Tp1`, typename `_Del`, typename = `_Convertible<typename unique_ptr<_Tp1, _Del>::pointer>>`
`shared_ptr` (std::unique_ptr< `_Tp1`, `_Del` > &&`__r`)
- constexpr `shared_ptr` (nullptr_t) noexcept
- template<typename `_Tp1` >
`shared_ptr` (std::auto_ptr< `_Tp1` > &&`__r`)
- `_Tp * get()` const noexcept
- `operator bool()` const
- std::add_lvalue_reference< `_Tp` >::type `operator*` () const noexcept
- `_Tp * operator->()` const noexcept
- `shared_ptr` & `operator=` (const `shared_ptr` &) noexcept=default
- template<typename `_Tp1` >
`shared_ptr` & `operator=` (const `shared_ptr`< `_Tp1` > &`__r`) noexcept
- `shared_ptr` & `operator=` (`shared_ptr` &&`__r`) noexcept
- template<class `_Tp1` >
`shared_ptr` & `operator=` (`shared_ptr`< `_Tp1` > &&`__r`) noexcept
- template<typename `_Tp1`, typename `_Del` >
`shared_ptr` & `operator=` (std::unique_ptr< `_Tp1`, `_Del` > &&`__r`)
- template<typename `_Tp1` >
bool `owner_before` (`__shared_ptr`< `_Tp1`, `_Lp` > const &`__rhs`) const

- `template<typename _Tp1 >`
`bool owner_before (__weak_ptr< _Tp1, _Lp > const &__rhs) const`
- `void reset () noexcept`
- `template<typename _Tp1 >`
`void reset (_Tp1 * __p)`
- `template<typename _Tp1, typename _Deleter >`
`void reset (_Tp1 * __p, _Deleter __d)`
- `template<typename _Tp1, typename _Deleter, typename _Alloc >`
`void reset (_Tp1 * __p, _Deleter __d, _Alloc __a)`
- `void swap (__shared_ptr< _Tp, _Lp > &__other) noexcept`
- `bool unique () const noexcept`
- `long use_count () const noexcept`

Friends

- `template<typename _Tp1, typename _Alloc, typename... _Args>`
`shared_ptr< _Tp1 > allocate_shared (const _Alloc &__a, _Args &&...__args)`
- `class weak_ptr< _Tp >`

5.970.1 Detailed Description

```
template<typename _Tp>
class std::shared_ptr< _Tp >
```

A smart pointer with reference-counted copy semantics.

The object pointed to is deleted when the last `shared_ptr` pointing to it is destroyed or reset.

Definition at line 93 of file `bits/shared_ptr.h`.

5.970.2 Constructor & Destructor Documentation

5.970.2.1 `template<typename _Tp> constexpr std::shared_ptr< _Tp >::shared_ptr () [inline], [noexcept]`

Construct an empty `shared_ptr`.

Postcondition

```
use_count()==0 && get()==0
```

Definition at line 104 of file `bits/shared_ptr.h`.

Referenced by `std::auto_ptr< _Tp >::auto_ptr()`, and `std::shared_ptr< _State >::shared_ptr()`.

5.970.2.2 `template<typename _Tp> template<typename _Tp1 > std::shared_ptr< _Tp >::shared_ptr (_Tp1 * __p)`
`[inline], [explicit]`

Construct a `shared_ptr` that owns the pointer `__p`.

Parameters

\leftrightarrow __p	A pointer that is convertible to element_type*.
--------------------------	---

Postcondition

use_count() == 1 && get() == __p

Exceptions

std::bad_alloc, in	which case delete __p is called.
--------------------	----------------------------------

Definition at line 116 of file bits/shared_ptr.h.

5.970.2.3 `template<typename _Tp> template<typename _Tp1, typename _Deleter> std::shared_ptr< _Tp >::shared_ptr (_Tp1 * __p, _Deleter __d) [inline]`

Construct a shared_ptr that owns the pointer __p and the deleter __d.

Parameters

\leftrightarrow __p	A pointer.
\leftrightarrow __d	A deleter.

Postcondition

use_count() == 1 && get() == __p

Exceptions

std::bad_alloc, in	which case __d(__p) is called.
--------------------	--------------------------------

Requirements: _Deleter's copy constructor and destructor must not throw

__shared_ptr will release __p by calling __d(__p)

Definition at line 133 of file bits/shared_ptr.h.

5.970.2.4 `template<typename _Tp> template<typename _Deleter> std::shared_ptr< _Tp >::shared_ptr (nullptr_t __p, _Deleter __d) [inline]`

Construct a shared_ptr that owns a null pointer and the deleter __d.

Parameters

\leftrightarrow __p	A null pointer constant.
\leftrightarrow __d	A deleter.

Postcondition

use_count() == 1 && get() == __p

Exceptions

<i>std::bad_alloc</i> , in	which case __d(__p) is called.
----------------------------	--------------------------------

Requirements: _Deleter's copy constructor and destructor must not throw

The last owner will call __d(__p)

Definition at line 150 of file bits/shared_ptr.h.

5.970.2.5 `template<typename _Tp> template<typename _Tp1, typename _Deleter, typename _Alloc > std::shared_ptr<_Tp>::shared_ptr (_Tp1 * __p, _Deleter __d, _Alloc __a) [inline]`

Construct a shared_ptr that owns the pointer __p and the deleter __d.

Parameters

\leftrightarrow __p	A pointer.
\leftrightarrow __d	A deleter.
\leftrightarrow __a	An allocator.

Postcondition

use_count() == 1 && get() == __p

Exceptions

<i>std::bad_alloc</i> , in	which case __d(__p) is called.
----------------------------	--------------------------------

Requirements: _Deleter's copy constructor and destructor must not throw _Alloc's copy constructor and destructor must not throw.

__shared_ptr will release __p by calling __d(__p)

Definition at line 169 of file bits/shared_ptr.h.

5.970.2.6 `template<typename _Tp> template<typename _Deleter, typename _Alloc > std::shared_ptr< _Tp >::shared_ptr (nullptr_t __p, _Deleter __d, _Alloc __a) [inline]`

Construct a shared_ptr that owns a null pointer and the deleter __d.

Parameters

<code>__p</code>	A null pointer constant.
<code>__d</code>	A deleter.
<code>__a</code>	An allocator.

Postcondition

`use_count() == 1 && get() == __p`

Exceptions

<code>std::bad_alloc</code> , in	which case __d(__p) is called.
----------------------------------	--------------------------------

Requirements: _Deleter's copy constructor and destructor must not throw _Alloc's copy constructor and destructor must not throw.

The last owner will call __d(__p)

Definition at line 188 of file bits/shared_ptr.h.

5.970.2.7 `template<typename _Tp> template<typename _Tp1 > std::shared_ptr< _Tp >::shared_ptr (const shared_ptr< _Tp1 > & __r, _Tp * __p) [inline], [noexcept]`

Constructs a shared_ptr instance that stores __p and shares ownership with __r.

Parameters

<code>__r</code>	A shared_ptr.
<code>__p</code>	A pointer that will remain valid while *__r is valid.

Postcondition

```
get() == __p && use_count() == __r.use_count()
```

This can be used to construct a `shared_ptr` to a sub-object of an object managed by an existing `shared_ptr`.

```
shared_ptr< pair<int,int> > pii(new pair<int,int>());
shared_ptr<int> pi(pii, &pii->first);
assert(pii.use_count() == 2);
```

Definition at line 210 of file `bits/shared_ptr.h`.

5.970.2.8 `template<typename _Tp> template<typename _Tp1, typename = _Convertible<_Tp1*>> std::shared_ptr<_Tp> >::shared_ptr (const shared_ptr<_Tp1> &__r) [inline], [noexcept]`

If `__r` is empty, constructs an empty `shared_ptr`; otherwise construct a `shared_ptr` that shares ownership with `__r`.

Parameters

↩	A shared_ptr.
↩	
↩	
↩	
<i>r</i>	

Postcondition

```
get() == __r.get() && use_count() == __r.use_count()
```

Definition at line 221 of file `bits/shared_ptr.h`.

5.970.2.9 `template<typename _Tp> std::shared_ptr<_Tp> >::shared_ptr (shared_ptr<_Tp> &&__r) [inline], [noexcept]`

Move-constructs a `shared_ptr` instance from `__r`.

Parameters

↩	A shared_ptr rvalue.
↩	
↩	
↩	
<i>r</i>	

Postcondition

*this contains the old value of `__r`, `__r` is empty.

Definition at line 229 of file `bits/shared_ptr.h`.

5.970.2.10 `template<typename _Tp> template<typename _Tp1, typename = _Convertible<_Tp1*>> std::shared_ptr<_Tp>::shared_ptr(shared_ptr<_Tp1> && __r) [inline], [noexcept]`

Move-constructs a `shared_ptr` instance from `__r`.

Parameters

<code>__r</code>	A <code>shared_ptr</code> rvalue.
------------------	-----------------------------------

Postcondition

*this contains the old value of `__r`, `__r` is empty.

Definition at line 238 of file `bits/shared_ptr.h`.

5.970.2.11 `template<typename _Tp> template<typename _Tp1> std::shared_ptr<_Tp>::shared_ptr(const weak_ptr<_Tp1> & __r) [inline], [explicit]`

Constructs a `shared_ptr` that shares ownership with `__r` and stores a copy of the pointer stored in `__r`.

Parameters

<code>__r</code>	A <code>weak_ptr</code> .
------------------	---------------------------

Postcondition

`use_count() == __r.use_count()`

Exceptions

<i>bad_weak_ptr</i>	when <code>__r.expired()</code> , in which case the constructor has no effect.
---------------------	--

Definition at line 250 of file `bits/shared_ptr.h`.

5.970.2.12 `template<typename _Tp> constexpr std::shared_ptr<_Tp>::shared_ptr(nullptr_t) [inline], [noexcept]`

Construct an empty `shared_ptr`.

Postcondition

`use_count() == 0 && get() == nullptr`

Definition at line 269 of file `bits/shared_ptr.h`.

5.970.3 Friends And Related Function Documentation

5.970.3.1 `template<typename _Tp> template<typename _Tp1, typename _Alloc, typename... _Args> shared_ptr<_Tp1>
allocate_shared (const _Alloc & __a, _Args &&... __args) [friend]`

Create an object that is owned by a `shared_ptr`.

Parameters

<code>__a</code>	An allocator.
<code>__args</code>	Arguments for the <code>_Tp</code> object's constructor.

Returns

A `shared_ptr` that owns the newly created object.

Exceptions

<i>An</i>	exception thrown from <code>_Alloc::allocate</code> or from the constructor of <code>_Tp</code> .
-----------	---

A copy of `__a` will be used to allocate memory for the `shared_ptr` and the new object.

Definition at line 617 of file `bits/shared_ptr.h`.

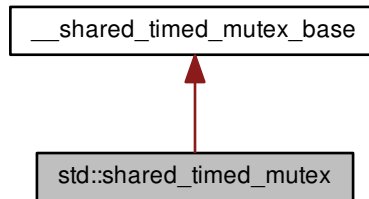
Referenced by `std::shared_ptr<_State>::shared_ptr()`.

The documentation for this class was generated from the following files:

- [bits/shared_ptr.h](#)
- [auto_ptr.h](#)

5.971 `std::shared_timed_mutex` Class Reference

Inheritance diagram for `std::shared_timed_mutex`:



Public Member Functions

- **`shared_timed_mutex`** (const [shared_timed_mutex](#) &)=delete
- void **`lock`** ()
- void **`lock_shared`** ()
- [shared_timed_mutex](#) & **`operator=`** (const [shared_timed_mutex](#) &)=delete
- bool **`try_lock`** ()
- template<typename `_Rep` , typename `_Period` >
bool **`try_lock_for`** (const [chrono::duration](#)< `_Rep`, `_Period` > &__rel_time)
- bool **`try_lock_shared`** ()
- template<typename `_Rep` , typename `_Period` >
bool **`try_lock_shared_for`** (const [chrono::duration](#)< `_Rep`, `_Period` > &__rel_time)
- template<typename `_Clock` , typename `_Duration` >
bool **`try_lock_shared_until`** (const [chrono::time_point](#)< `_Clock`, `_Duration` > &__abs_time)
- template<typename `_Clock` , typename `_Duration` >
bool **`try_lock_until`** (const [chrono::time_point](#)< `_Clock`, `_Duration` > &__abs_time)
- void **`unlock`** ()
- void **`unlock_shared`** ()

5.971.1 Detailed Description

The standard shared timed mutex type.

Definition at line 361 of file `shared_mutex`.

The documentation for this class was generated from the following file:

- [shared_mutex](#)

5.972 `std::shuffle_order_engine<_RandomNumberEngine, __k>` Class Template Reference

Public Types

- typedef `_RandomNumberEngine::result_type` [result_type](#)

Public Member Functions

- [shuffle_order_engine](#) ()
- [shuffle_order_engine](#) (const `_RandomNumberEngine` &__rng)
- [shuffle_order_engine](#) (`_RandomNumberEngine` &&__rng)
- [shuffle_order_engine](#) ([result_type](#) __s)
- template<typename `_Sseq` , typename = typename `std::enable_if<!std::is_same<_Sseq, shuffle_order_engine>::value && !std::is_↵`
same<_Sseq, `_RandomNumberEngine`>::value> ::type>
[shuffle_order_engine](#) (`_Sseq` &__q)
- const `_RandomNumberEngine` & [base](#) () const noexcept
- void [discard](#) (unsigned long long __z)
- [result_type](#) [operator](#)() ()
- void [seed](#) ()
- void [seed](#) ([result_type](#) __s)
- template<typename `_Sseq` >
void [seed](#) (`_Sseq` &__q)

Static Public Member Functions

- static constexpr [result_type](#) [max](#) ()
- static constexpr [result_type](#) [min](#) ()

Static Public Attributes

- static constexpr size_t [table_size](#)

Friends

- template<typename `_RandomNumberEngine1` , size_t __k1, typename `_CharT` , typename `_Traits` >
[std::basic_ostream](#)< `_CharT`, `_Traits` > & [operator<<](#) ([std::basic_ostream](#)< `_CharT`, `_Traits` > &__os, const
`std::shuffle_order_engine`< `_RandomNumberEngine1`, __k1 > &__x)
- bool [operator==](#) (const [shuffle_order_engine](#) &__lhs, const [shuffle_order_engine](#) &__rhs)
- template<typename `_RandomNumberEngine1` , size_t __k1, typename `_CharT` , typename `_Traits` >
[std::basic_istream](#)< `_CharT`, `_Traits` > & [operator>>](#) ([std::basic_istream](#)< `_CharT`, `_Traits` > &__is, [std::↵](#)
`std::shuffle_order_engine`< `_RandomNumberEngine1`, __k1 > &__x)

5.972.1 Detailed Description

```
template<typename _RandomNumberEngine, size_t __k>
class std::shuffle_order_engine<_RandomNumberEngine, __k>
```

Produces random numbers by combining random numbers from some base engine to produce random numbers with a specifies number of bits `__w`.

Definition at line 1284 of file random.h.

5.972.2 Member Typedef Documentation

```
5.972.2.1 template<typename _RandomNumberEngine, size_t __k> typedef _RandomNumberEngine::result_type
std::shuffle_order_engine<_RandomNumberEngine, __k>::result_type
```

The type of the generated random value.

Definition at line 1287 of file random.h.

5.972.3 Constructor & Destructor Documentation

```
5.972.3.1 template<typename _RandomNumberEngine, size_t __k> std::shuffle_order_engine<_RandomNumberEngine,
__k>::shuffle_order_engine ( ) [inline]
```

Constructs a default shuffle_order_engine engine.

The underlying engine is default constructed as well.

Definition at line 1300 of file random.h.

```
5.972.3.2 template<typename _RandomNumberEngine, size_t __k> std::shuffle_order_engine<_RandomNumberEngine,
__k>::shuffle_order_engine ( const _RandomNumberEngine & __rng ) [inline], [explicit]
```

Copy constructs a shuffle_order_engine engine.

Copies an existing base class random number generator.

Parameters

<code>__rng</code>	An existing (base class) engine object.
--------------------	---

Definition at line 1311 of file random.h.

```
5.972.3.3 template<typename _RandomNumberEngine, size_t __k> std::shuffle_order_engine<_RandomNumberEngine,
__k>::shuffle_order_engine ( _RandomNumberEngine && __rng ) [inline], [explicit]
```

Move constructs a shuffle_order_engine engine.

Copies an existing base class random number generator.

Parameters

<code>__rng</code>	An existing (base class) engine object.
--------------------	---

Definition at line 1322 of file random.h.

5.972.3.4 `template<typename _RandomNumberEngine, size_t __k> std::shuffle_order_engine< _RandomNumberEngine, __k >::shuffle_order_engine (result_type __s) [inline],[explicit]`

Seed constructs a shuffle_order_engine engine.

Constructs the underlying generator engine seeded with `__s`.

Parameters

<code>__s</code>	A seed value for the base class engine.
------------------	---

Definition at line 1333 of file random.h.

5.972.3.5 `template<typename _RandomNumberEngine, size_t __k> template<typename _Sseq, typename = typename std::enable_if<!std::is_same<_Sseq, shuffle_order_engine>::value && !std::is_same<_Sseq, _RandomNumberEngine>::value> ::type> std::shuffle_order_engine< _RandomNumberEngine, __k >::shuffle_order_engine (_Sseq & __q) [inline],[explicit]`

Generator construct a shuffle_order_engine engine.

Parameters

<code>__q</code>	A seed sequence.
------------------	------------------

Definition at line 1347 of file random.h.

5.972.4 Member Function Documentation

5.972.4.1 `template<typename _RandomNumberEngine, size_t __k> const _RandomNumberEngine& std::shuffle_order_engine< _RandomNumberEngine, __k >::base () const [inline],[noexcept]`

Gets a const reference to the underlying generator engine object.

Definition at line 1390 of file random.h.

5.972.4.2 `template<typename _RandomNumberEngine, size_t __k> void std::shuffle_order_engine< _RandomNumberEngine, __k >::discard (unsigned long long __z) [inline]`

Discard a sequence of random numbers.

Definition at line 1411 of file random.h.

5.972.4.3 `template<typename _RandomNumberEngine, size_t __k> static constexpr result_type std::shuffle_order_engine< _RandomNumberEngine, __k >::max () [inline],[static]`

Gets the maximum value in the generated random number range.

Definition at line 1404 of file random.h.

References std::max().

5.972.4.4 `template<typename _RandomNumberEngine, size_t __k> static constexpr result_type std::shuffle_order_engine< _RandomNumberEngine, __k >::min () [inline],[static]`

Gets the minimum value in the generated random number range.

Definition at line 1397 of file random.h.

References std::min().

5.972.4.5 `template<typename _RandomNumberEngine, size_t __k> shuffle_order_engine< _RandomNumberEngine, __k >::result_type std::shuffle_order_engine< _RandomNumberEngine, __k >::operator() ()`

Gets the next value in the generated random number sequence.

Definition at line 818 of file bits/random.tcc.

References std::discard_block_engine< _RandomNumberEngine, __p, __r >::base(), std::dec(), std::fixed(), std::ios<← _base::flags(), std::left(), std::__detail::operator>>(), and std::skipws().

Referenced by std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >::operator()().

5.972.4.6 `template<typename _RandomNumberEngine, size_t __k> void std::shuffle_order_engine< _RandomNumberEngine, __k >::seed () [inline]`

Reseeds the shuffle_order_engine object with the default seed for the underlying base class generator engine.

Definition at line 1356 of file random.h.

5.972.4.7 `template<typename _RandomNumberEngine, size_t __k> void std::shuffle_order_engine< _RandomNumberEngine, __k >::seed (result_type __s) [inline]`

Reseeds the shuffle_order_engine object with the default seed for the underlying base class generator engine.

Definition at line 1367 of file random.h.

5.972.4.8 `template<typename _RandomNumberEngine, size_t __k> template<typename _Sseq > void std::shuffle_order_engine< _RandomNumberEngine, __k >::seed (_Sseq & __q) [inline]`

Reseeds the shuffle_order_engine object with the given seed sequence.

Parameters

<code>_↔</code>	A seed generator function.
<code>_q</code>	

Definition at line 1380 of file random.h.

5.972.5 Friends And Related Function Documentation

5.972.5.1 `template<typename _RandomNumberEngine, size_t __k> template<typename _RandomNumberEngine1 ,
size_t __k1, typename _CharT , typename _Traits > std::basic_ostream<_CharT, _Traits>& operator<< (
std::basic_ostream<_CharT, _Traits > &__os, const std::shuffle_order_engine<_RandomNumberEngine1,
__k1> &__x) [friend]`

Inserts the current state of a shuffle_order_engine random number generator engine `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A shuffle_order_engine random number generator engine.

Returns

The output stream with the state of `__x` inserted or in an error state.

5.972.5.2 `template<typename _RandomNumberEngine, size_t __k> bool operator==(const shuffle_order_engine<
_RandomNumberEngine, __k > &__lhs, const shuffle_order_engine<_RandomNumberEngine, __k > &__rhs)
[friend]`

Compares two shuffle_order_engine random number generator objects of the same type for equality.

Parameters

<code>__lhs</code>	A shuffle_order_engine random number generator object.
<code>__rhs</code>	Another shuffle_order_engine random number generator object.

Returns



true if the infinite sequences of generated values would be equal, false otherwise.

Definition at line 1435 of file random.h.

```
5.972.5.3 template<typename _RandomNumberEngine, size_t __k> template<typename _RandomNumberEngine1,
size_t __k1, typename _CharT, typename _Traits > std::basic_istream<_CharT, _Traits>& operator>> (
std::basic_istream<_CharT, _Traits > & __is, std::shuffle_order_engine<_RandomNumberEngine1, __k1 > &
__x ) [friend]
```

Extracts the current state of a % subtract_with_carry_engine random number generator engine __x from the input stream __is.

Parameters

 __is	An input stream.
 __x	A shuffle_order_engine random number generator engine.

Returns

The input stream with the state of __x extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.973 std::slice Class Reference

Public Member Functions

- [slice](#) ()
- [slice](#) (size_t __o, size_t __d, size_t __s)
- size_t [size](#) () const
- size_t [start](#) () const
- size_t [stride](#) () const

5.973.1 Detailed Description

Class defining one-dimensional subset of an array.

The slice class represents a one-dimensional subset of an array, specified by three parameters: start offset, size, and stride. The start offset is the index of the first element of the array that is part of the subset. The size is the total number of elements in the subset. Stride is the distance between each successive array element to include in the subset.

For example, with an array of size 10, and a slice with offset 1, size 3 and stride 2, the subset consists of array elements 1, 3, and 5.

Definition at line 59 of file slice_array.h.

The documentation for this class was generated from the following file:

- [slice_array.h](#)

5.974 `std::slice_array<_Tp>` Class Template Reference

Public Types

- typedef `_Tp` **value_type**

Public Member Functions

- `slice_array` (const `slice_array` &)
- void `operator%=>` (const `valarray`<_Tp> &) const
- template<class _Dom >
void `operator%=>` (const _Expr<_Dom, _Tp> &) const
- void `operator&=>` (const `valarray`<_Tp> &) const
- template<class _Dom >
void `operator&=>` (const _Expr<_Dom, _Tp> &) const
- void `operator*=>` (const `valarray`<_Tp> &) const
- template<class _Dom >
void `operator*=>` (const _Expr<_Dom, _Tp> &) const
- void `operator+=>` (const `valarray`<_Tp> &) const
- template<class _Dom >
void `operator+=>` (const _Expr<_Dom, _Tp> &) const
- void `operator-=>` (const `valarray`<_Tp> &) const
- template<class _Dom >
void `operator-=>` (const _Expr<_Dom, _Tp> &) const
- void `operator/=` (const `valarray`<_Tp> &) const
- template<class _Dom >
void `operator/=` (const _Expr<_Dom, _Tp> &) const
- void `operator<=>` (const `valarray`<_Tp> &) const
- template<class _Dom >
void `operator<=>` (const _Expr<_Dom, _Tp> &) const
- `slice_array` & `operator=` (const `slice_array` &)
- void `operator=` (const `valarray`<_Tp> &) const
- void `operator=` (const _Tp &) const
- template<class _Dom >
void `operator=` (const _Expr<_Dom, _Tp> &) const
- void `operator>=>` (const `valarray`<_Tp> &) const
- template<class _Dom >
void `operator>=>` (const _Expr<_Dom, _Tp> &) const
- void `operator^=>` (const `valarray`<_Tp> &) const
- template<class _Dom >
void `operator^=>` (const _Expr<_Dom, _Tp> &) const
- void `operator|=` (const `valarray`<_Tp> &) const
- template<class _Dom >
void `operator|=` (const _Expr<_Dom, _Tp> &) const

Friends

- class `valarray`<_Tp>

5.974.1 Detailed Description

```
template<typename _Tp>
class std::slice_array<_Tp>
```

Reference to one-dimensional subset of an array.

A `slice_array` is a reference to the actual elements of an array specified by a slice. The way to get a `slice_array` is to call `operator[]`(slice) on a valarray. The returned `slice_array` then permits carrying operations out on the referenced subset of elements in the original valarray. For example, `operator+=(valarray)` will add values to the subset of elements in the underlying valarray this `slice_array` refers to.

Parameters

<i>Tp</i>	Element type.
-----------	---------------

Definition at line 80 of file `valarray`.

The documentation for this class was generated from the following files:

- [valarray](#)
- [slice_array.h](#)

5.975 `std::stack<_Tp, _Sequence>` Class Template Reference

Public Types

- typedef `_Sequence::const_reference` **const_reference**
- typedef `_Sequence` **container_type**
- typedef `_Sequence::reference` **reference**
- typedef `_Sequence::size_type` **size_type**
- typedef `_Sequence::value_type` **value_type**

Public Member Functions

- [stack](#) (const `_Sequence` &__c)
- **stack** (`_Sequence` &&__c=`_Sequence`())
- template<typename `_Alloc`, typename `_Requires` = `_Uses<_Alloc>>`
stack (const `_Alloc` &__a)
- template<typename `_Alloc`, typename `_Requires` = `_Uses<_Alloc>>`
stack (const `_Sequence` &__c, const `_Alloc` &__a)
- template<typename `_Alloc`, typename `_Requires` = `_Uses<_Alloc>>`
stack (`_Sequence` &&__c, const `_Alloc` &__a)
- template<typename `_Alloc`, typename `_Requires` = `_Uses<_Alloc>>`
stack (const [stack](#) &__q, const `_Alloc` &__a)
- template<typename `_Alloc`, typename `_Requires` = `_Uses<_Alloc>>`
stack ([stack](#) &&__q, const `_Alloc` &__a)

- `template<typename... _Args>`
`void emplace (_Args &&... __args)`
- `bool empty () const`
- `void pop ()`
- `void push (const value_type & __x)`
- `void push (value_type && __x)`
- `size_type size () const`
- `void swap (stack & __s) noexcept(__is_nothrow_swappable< _Tp >::value)`
- `reference top ()`
- `const_reference top () const`

Protected Attributes

- `_Sequence c`

Friends

- `template<typename _Tp1, typename _Seq1 >`
`bool operator< (const stack< _Tp1, _Seq1 > &, const stack< _Tp1, _Seq1 > &)`
- `template<typename _Tp1, typename _Seq1 >`
`bool operator== (const stack< _Tp1, _Seq1 > &, const stack< _Tp1, _Seq1 > &)`

5.975.1 Detailed Description

```
template<typename _Tp, typename _Sequence = deque<_Tp>>
class std::stack< _Tp, _Sequence >
```

A standard container giving FILO behavior.

Template Parameters

<code>_Tp</code>	Type of element.
<code>_Sequence</code>	Type of underlying sequence, defaults to <code>deque<_Tp></code> .

Meets many of the requirements of a `container`, but does not define anything to do with iterators. Very few of the other standard container interfaces are defined.

This is not a true container, but an *adaptor*. It holds another container, and provides a wrapper interface to that container. The wrapper is what enforces strict first-in-last-out stack behavior.

The second template parameter defines the type of the underlying sequence/container. It defaults to `std::deque`, but it can be any type that supports `back`, `push_back`, and `pop_front`, such as `std::list`, `std::vector`, or an appropriate user-defined type.

Members not found in *normal* containers are `container_type`, which is a typedef for the second Sequence parameter, and `push`, `pop`, and `top`, which are standard stack/FILO operations.

Definition at line 99 of file `stl_stack.h`.

5.975.2 Constructor & Destructor Documentation

5.975.2.1 `template<typename _Tp, typename _Sequence = deque<_Tp>> std::stack<_Tp, _Sequence>::stack (const _Sequence &__c) [inline], [explicit]`

Default constructor creates no elements.

Definition at line 145 of file `stl_stack.h`.

Referenced by `std::stack<_StateSeqT>::stack()`.

5.975.3 Member Function Documentation

5.975.3.1 `template<typename _Tp, typename _Sequence = deque<_Tp>> bool std::stack<_Tp, _Sequence>::empty () const [inline]`

Returns true if the stack is empty.

Definition at line 178 of file `stl_stack.h`.

5.975.3.2 `template<typename _Tp, typename _Sequence = deque<_Tp>> void std::stack<_Tp, _Sequence>::pop () [inline]`

Removes first element.

This is a typical stack operation. It shrinks the stack by one. The time complexity of the operation depends on the underlying sequence.

Note that no data is returned, and if the first element's data is needed, it should be retrieved before `pop()` is called.

Definition at line 244 of file `stl_stack.h`.

5.975.3.3 `template<typename _Tp, typename _Sequence = deque<_Tp>> void std::stack<_Tp, _Sequence>::push (const value_type &__x) [inline]`

Add data to the top of the stack.

Parameters

<code>__x</code>	Data to be added.
------------------	-------------------

This is a typical stack operation. The function creates an element at the top of the stack and assigns the given data to it. The time complexity of the operation depends on the underlying sequence.

Definition at line 218 of file `stl_stack.h`.

Referenced by `std::stack<_StateSeqT>::push()`.

5.975.3.4 `template<typename _Tp, typename _Sequence = deque<_Tp>> size_type std::stack<_Tp, _Sequence>::size ()`
`const [inline]`

Returns the number of elements in the stack.

Definition at line 183 of file `stl_stack.h`.

5.975.3.5 `template<typename _Tp, typename _Sequence = deque<_Tp>> reference std::stack<_Tp, _Sequence>::top ()`
`[inline]`

Returns a read/write reference to the data at the first element of the stack.

Definition at line 191 of file `stl_stack.h`.

5.975.3.6 `template<typename _Tp, typename _Sequence = deque<_Tp>> const_reference std::stack<_Tp, _Sequence>::top`
`() const [inline]`

Returns a read-only (constant) reference to the data at the first element of the stack.

Definition at line 202 of file `stl_stack.h`.

The documentation for this class was generated from the following file:

- [stl_stack.h](#)

5.976 `std::student_t_distribution<_RealType>` Class Template Reference

Classes

- struct [param_type](#)

Public Types

- typedef `_RealType` [result_type](#)

Public Member Functions

- **student_t_distribution** (_RealType __n=_RealType(1))
- **student_t_distribution** (const [param_type](#) &__p)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **__generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **__generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- template<typename _UniformRandomNumberGenerator >
void **__generate** ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
void **__generate** ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [result_type](#) **max** () const
- [result_type](#) **min** () const
- _RealType **n** () const
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [param_type](#) **param** () const
- void **param** (const [param_type](#) &__param)
- void **reset** ()

Friends

- template<typename _RealType1, typename _CharT, typename _Traits >
[std::basic_ostream](#)< _CharT, _Traits > & **operator<<** ([std::basic_ostream](#)< _CharT, _Traits > &__os, const [std::student_t_distribution](#)< _RealType1 > &__x)
- bool **operator==** (const [student_t_distribution](#) &__d1, const [student_t_distribution](#) &__d2)
- template<typename _RealType1, typename _CharT, typename _Traits >
[std::basic_istream](#)< _CharT, _Traits > & **operator>>** ([std::basic_istream](#)< _CharT, _Traits > &__is, [std::student_t_distribution](#)< _RealType1 > &__x)

5.976.1 Detailed Description

```
template<typename _RealType = double>
class std::student_t_distribution<_RealType>
```

A [student_t_distribution](#) random number distribution.

The formula for the normal probability mass function is:

$$p(x|n) = \frac{1}{\sqrt{(n\pi)}} \frac{\Gamma((n+1)/2)}{\Gamma(n/2)} \left(1 + \frac{x^2}{n}\right)^{-(n+1)/2}$$

Definition at line 3189 of file random.h.

5.976.2 Member Typedef Documentation

5.976.2.1 `template<typename _RealType = double> typedef _RealType std::student_t_distribution< _RealType >::result_type`

The type of the range of the distribution.

Definition at line 3192 of file random.h.

5.976.3 Member Function Documentation

5.976.3.1 `template<typename _RealType = double> result_type std::student_t_distribution< _RealType >::max () const [inline]`

Returns the least upper bound value of the distribution.

Definition at line 3272 of file random.h.

References `std::numeric_limits< _Tp >::max()`.

5.976.3.2 `template<typename _RealType = double> result_type std::student_t_distribution< _RealType >::min () const [inline]`

Returns the greatest lower bound value of the distribution.

Definition at line 3265 of file random.h.

References `std::numeric_limits< _Tp >::lowest()`.

5.976.3.3 `template<typename _RealType = double> template<typename _UniformRandomNumberGenerator > result_type std::student_t_distribution< _RealType >::operator() (_UniformRandomNumberGenerator & __urng) [inline]`

Generating functions.

Definition at line 3280 of file random.h.

References `std::sqrt()`.

5.976.3.4 `template<typename _RealType = double> param_type std::student_t_distribution< _RealType >::param () const [inline]`

Returns the parameter set of the distribution.

Definition at line 3250 of file random.h.

Referenced by `std::operator>>()`.

5.976.3.5 `template<typename _RealType = double> void std::student_t_distribution< _RealType >::param (const param_type & __param) [inline]`

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

Definition at line 3258 of file random.h.

5.976.3.6 `template<typename _RealType = double> void std::student_t_distribution<_RealType>::reset ()`
`[inline]`

Resets the distribution state.

Definition at line 3233 of file random.h.

5.976.4 Friends And Related Function Documentation

5.976.4.1 `template<typename _RealType = double> template<typename _RealType1, typename _CharT, typename _Traits>`
`std::basic_ostream<_CharT, _Traits>& operator<< (std::basic_ostream<_CharT, _Traits> & __os, const`
`std::student_t_distribution<_RealType1> & __x) [friend]`

Inserts a student_t_distribution random number distribution `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A student_t_distribution random number distribution.

Returns

The output stream with the state of `__x` inserted or in an error state.

5.976.4.2 `template<typename _RealType = double> bool operator==(const student_t_distribution<_RealType> & __d1,`
`const student_t_distribution<_RealType> & __d2) [friend]`

Return true if two Student t distributions have the same parameters and the sequences that would be generated are equal.

Definition at line 3329 of file random.h.

5.976.4.3 `template<typename _RealType = double> template<typename _RealType1, typename _CharT, typename _Traits>`
`> std::basic_istream<_CharT, _Traits>& operator>> (std::basic_istream<_CharT, _Traits> & __is,`
`std::student_t_distribution<_RealType1> & __x) [friend]`

Extracts a student_t_distribution random number distribution `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A <code>student_t_distribution</code> random number generator engine.

Returns

The input stream with `__x` extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.977 `std::student_t_distribution<_RealType>::param_type` Struct Reference

Public Types

- typedef `student_t_distribution<_RealType>` **distribution_type**

Public Member Functions

- **param_type** (`_RealType __n=_RealType(1)`)
- `_RealType n` () const

Friends

- bool **operator==** (const `param_type` &__p1, const `param_type` &__p2)

5.977.1 Detailed Description

```
template<typename _RealType = double>
struct std::student_t_distribution<_RealType>::param_type
```

Parameter type.

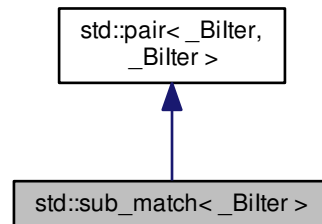
Definition at line 3198 of file `random.h`.

The documentation for this struct was generated from the following file:

- [random.h](#)

5.978 `std::sub_match<_Bilter>` Class Template Reference

Inheritance diagram for `std::sub_match<_Bilter>`:



Public Types

- using `_PCCFP` = `_PCC<lis_same<_Bilter, _U1>::value||lis_same<_Bilter, _U2>::value, _Bilter, _Bilter>`
- using `_PCCP` = `_PCC< true, _Bilter, _Bilter>`
- typedef `__iter_traits::difference_type` **difference_type**
- typedef `_Bilter` **first_type**
- typedef `_Bilter` **iterator**
- typedef `_Bilter` **second_type**
- typedef `std::basic_string< value_type>` **string_type**
- typedef `__iter_traits::value_type` **value_type**

Public Member Functions

- int `compare` (const `sub_match` &__s) const
- int `compare` (const `string_type` &__s) const
- int `compare` (const `value_type` *__s) const
- `difference_type` `length` () const
- `operator string_type` () const
- `string_type` `str` () const
- void `swap` (`pair` &__p) noexcept(`__is_nothrow_swappable<_Bilter>::value` &&`__is_nothrow_swappable<_Bilter>::value`)

Public Attributes

- `_Bilter` **first**
- bool **matched**
- `_Bilter` **second**

5.978.1 Detailed Description

```
template<typename _Bilter>
class std::sub_match< _Bilter >
```

A sequence of characters matched by a particular marked sub-expression.

An object of this class is essentially a pair of iterators marking a matched subexpression within a regular expression pattern match. Such objects can be converted to and compared with `std::basic_string` objects of a similar base character type as the pattern matched by the regular expression.

The iterators that make up the pair are the usual half-open interval referencing the actual original pattern matched.

Definition at line 824 of file `regex.h`.

5.978.2 Member Typedef Documentation

5.978.2.1 `using std::pair< _Bilter , _Bilter >::_PCCFP = _PCC<!is_same<_Bilter , _U1>::value || !is_same<_Bilter , _U2>::value, _Bilter , _Bilter > [inherited]`

There is also a templated copy ctor for the `pair` class itself.

Definition at line 264 of file `stl_pair.h`.

5.978.2.2 `using std::pair< _Bilter , _Bilter >::_PCCP = _PCC<true, _Bilter , _Bilter > [inherited]`

Two objects may be passed to a `pair` constructor to be copied.

Definition at line 233 of file `stl_pair.h`.

5.978.2.3 `typedef _Bilter std::pair< _Bilter , _Bilter >::second_type [inherited]`

`first_type` is the first bound type

Definition at line 193 of file `stl_pair.h`.

5.978.3 Member Function Documentation

5.978.3.1 `template<typename _Bilter> int std::sub_match< _Bilter >::compare (const sub_match< _Bilter > & __s) const [inline]`

Compares this and another matched sequence.

Parameters

<code>__s</code>	Another matched sequence to compare to this one.
------------------	--

Return values

<0	this matched sequence will collate before __s.
=0	this matched sequence is equivalent to __s.
>0	this matched sequence will collate after __s.


Definition at line 885 of file regex.h.

Referenced by std::operator!=(), std::operator<(), std::operator<=(), std::operator==(), std::operator>(), and std::operator>=().

5.978.3.2 template<typename _Bilter> int std::sub_match<_Bilter>::compare (const string_type & __s) const
[inline]

Compares this sub_match to a string.

Parameters

 __s	A string to compare to this sub_match.
---	--

Return values


<0	this matched sequence will collate before __s.
=0	this matched sequence is equivalent to __s.
>0	this matched sequence will collate after __s.

Definition at line 898 of file regex.h.

5.978.3.3 template<typename _Bilter> int std::sub_match<_Bilter>::compare (const value_type * __s) const
[inline]

Compares this sub_match to a C-style string.

Parameters

 __s	A C-style string to compare to this sub_match.
---	--

Return values

<0	this matched sequence will collate before __s.
=0	this matched sequence is equivalent to __s.
>0	this matched sequence will collate after __s.

Definition at line 911 of file regex.h.

5.978.3.4 `template<typename _Bilter> difference_type std::sub_match<_Bilter>::length () const [inline]`

Gets the length of the matching sequence.

Definition at line 842 of file regex.h.

5.978.3.5 `template<typename _Bilter> std::sub_match<_Bilter>::operator string_type () const [inline]`

Gets the matching sequence as a string.

Returns

the matching sequence as a string.

This is the implicit conversion operator. It is identical to the `str()` member function except that it will want to pop up in unexpected places and cause a great deal of confusion and cursing from the unwary.

Definition at line 855 of file regex.h.

5.978.3.6 `template<typename _Bilter> string_type std::sub_match<_Bilter>::str () const [inline]`

Gets the matching sequence as a string.

Returns

the matching sequence as a string.

Definition at line 868 of file regex.h.

Referenced by `std::sub_match<_Bi_iter>::compare()`, and `std::operator<<()`.

5.978.4 Member Data Documentation

5.978.4.1 `_Bilter std::pair<_Bilter, _Bilter>::first [inherited]`

`second_type` is the second bound type

Definition at line 195 of file `stl_pair.h`.

5.978.4.2 `_Bilter std::pair<_Bilter, _Bilter>::second [inherited]`

`first` is a copy of the first object

Definition at line 196 of file `stl_pair.h`.

The documentation for this class was generated from the following file:

- [regex.h](#)

5.979 std::subtract_with_carry_engine< _UIntType, __w, __s, __r > Class Template Reference

Public Types

- typedef _UIntType [result_type](#)

Public Member Functions

- [subtract_with_carry_engine](#) ([result_type](#) __sd=default_seed)
- template<typename _Sseq, typename = typename std::enable_if<!std::is_same<_Sseq, subtract_with_carry_engine>::value>::type>
[subtract_with_carry_engine](#) (_Sseq &__q)
- void [discard](#) (unsigned long long __z)
- [result_type](#) operator() ()
- void [seed](#) ([result_type](#) __sd=default_seed)
- template<typename _Sseq >
std::enable_if< [std::is_class](#)< _Sseq >::value >::type [seed](#) (_Sseq &__q)

Static Public Member Functions

- static constexpr [result_type](#) [max](#) ()
- static constexpr [result_type](#) [min](#) ()

Static Public Attributes

- static constexpr [result_type](#) [default_seed](#)
- static constexpr size_t [long_lag](#)
- static constexpr size_t [short_lag](#)
- static constexpr size_t [word_size](#)

Friends

- template<typename _UIntType1, size_t __w1, size_t __s1, size_t __r1, typename _CharT, typename _Traits >
[std::basic_ostream](#)< _CharT, _Traits > & [operator<<](#) ([std::basic_ostream](#)< _CharT, _Traits > &__os, const
[std::subtract_with_carry_engine](#)< _UIntType1, __w1, __s1, __r1 > &__x)
- bool [operator==](#) (const [subtract_with_carry_engine](#) &__lhs, const [subtract_with_carry_engine](#) &__rhs)
- template<typename _UIntType1, size_t __w1, size_t __s1, size_t __r1, typename _CharT, typename _Traits >
[std::basic_istream](#)< _CharT, _Traits > & [operator>>](#) ([std::basic_istream](#)< _CharT, _Traits > &__is, [std::subtract_with_carry_engine](#)< _UIntType1, __w1, __s1, __r1 > &__x)

5.979.1 Detailed Description

```
template<typename _UIntType, size_t __w, size_t __s, size_t __r>
class std::subtract_with_carry_engine< _UIntType, __w, __s, __r >
```

The Marsaglia-Zaman generator.

This is a model of a Generalized Fibonacci discrete random number generator, sometimes referred to as the SWC generator.

A discrete random number generator that produces pseudorandom numbers using:

$$x_i \leftarrow (x_{i-s} - x_{i-r} - carry_{i-1}) \bmod m$$

The size of the state is r and the maximum period of the generator is $(m^r - m^s - 1)$.

Definition at line 659 of file random.h.

5.979.2 Member Typedef Documentation

```
5.979.2.1 template<typename _UIntType, size_t __w, size_t __s, size_t __r> typedef _UIntType
std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::result_type
```

The type of the generated random value.

Definition at line 662 of file random.h.

5.979.3 Constructor & Destructor Documentation

```
5.979.3.1 template<typename _UIntType, size_t __w, size_t __s, size_t __r> std::subtract_with_carry_engine< _UIntType,
__w, __s, __r >::subtract_with_carry_engine ( result_type __sd=default_seed ) [inline],
[explicit]
```

Constructs an explicitly seeded % subtract_with_carry_engine random number generator.

Definition at line 683 of file random.h.

```
5.979.3.2 template<typename _UIntType, size_t __w, size_t __s, size_t __r> template<typename _Sseq, typename
= typename std::enable_if<!std::is_same<_Sseq, subtract_with_carry_engine>::value> ::type>
std::subtract_with_carry_engine< _UIntType, __w, __s, __r >::subtract_with_carry_engine ( _Sseq & __q )
[inline], [explicit]
```

Constructs a subtract_with_carry_engine random number engine seeded from the seed sequence __q.

Parameters

$_q$	the seed sequence.
-------	--------------------

Definition at line 696 of file random.h.

5.979.4 Member Function Documentation

5.979.4.1 `template<typename _UIntType, size_t __w, size_t __s, size_t __r> void std::subtract_with_carry_engine<_UIntType, __w, __s, __r>::discard(unsigned long long __z) [inline]`

Discard a sequence of random numbers.

Definition at line 742 of file random.h.

5.979.4.2 `template<typename _UIntType, size_t __w, size_t __s, size_t __r> static constexpr result_type std::subtract_with_carry_engine<_UIntType, __w, __s, __r>::max() [inline],[static]`

Gets the inclusive maximum value of the range of random integers returned by this generator.

Definition at line 735 of file random.h.

5.979.4.3 `template<typename _UIntType, size_t __w, size_t __s, size_t __r> static constexpr result_type std::subtract_with_carry_engine<_UIntType, __w, __s, __r>::min() [inline],[static]`

Gets the inclusive minimum value of the range of random integers returned by this generator.

Definition at line 727 of file random.h.

5.979.4.4 `template<typename _UIntType, size_t __w, size_t __s, size_t __r> subtract_with_carry_engine<_UIntType, __w, __s, __r>::result_type std::subtract_with_carry_engine<_UIntType, __w, __s, __r>::operator()()`

Gets the next random number in the sequence.

Definition at line 598 of file bits/random.tcc.

References `std::dec()`, `std::fixed()`, `std::ios_base::flags()`, `std::left()`, `std::discard_block_engine<_RandomNumberEngine, __p, __r>::operator()()`, `std::__detail::operator>>()`, and `std::skipws()`.

Referenced by `std::subtract_with_carry_engine<_UIntType, __w, __s, __r>::seed()`.

5.979.4.5 `template<typename _UIntType, size_t __w, size_t __s, size_t __r> void std::subtract_with_carry_engine<_UIntType, __w, __s, __r>::seed (result_type __sd=default_seed)`

Seeds the initial state x_0 of the random number generator.

N1688[4.19] modifies this as follows. If `__value == 0`, sets value to 19780503. In any case, with a linear congruential generator $lcg(i)$ having parameters $m_{lcg} = 2147483563$, $a_{lcg} = 40014$, $c_{lcg} = 0$, and $lcg(0) = value$, sets $x_{-r} \dots x_{-1}$ to $lcg(1) \bmod m \dots lcg(r) \bmod m$ respectively. If $x_{-1} = 0$ set carry to 1, otherwise sets carry to 0.

Definition at line 543 of file `bits/random.tcc`.

Referenced by `std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f>::discard()`.

5.979.4.6 `template<typename _UIntType, size_t __w, size_t __s, size_t __r> template<typename _Sseq> std::enable_if<std::is_class<_Sseq>::value>::type std::subtract_with_carry_engine<_UIntType, __w, __s, __r>::seed (_Sseq & __q)`

Seeds the initial state x_0 of the % `subtract_with_carry_engine` random number generator.

Definition at line 572 of file `bits/random.tcc`.

References `std::subtract_with_carry_engine< _UIntType, __w, __s, __r>::operator()()`.

5.979.5 Friends And Related Function Documentation

5.979.5.1 `template<typename _UIntType, size_t __w, size_t __s, size_t __r> template<typename _UIntType1, size_t __w1, size_t __s1, size_t __r1, typename _CharT, typename _Traits> std::basic_ostream<_CharT, _Traits> & operator<< (std::basic_ostream<_CharT, _Traits> & __os, const std::subtract_with_carry_engine<_UIntType1, __w1, __s1, __r1> & __x) [friend]`

Inserts the current state of a % `subtract_with_carry_engine` random number generator engine `__x` into the output stream `__os`.

Parameters

<code>__os</code>	An output stream.
<code>__x</code>	A % <code>subtract_with_carry_engine</code> random number generator engine.

Returns

The output stream with the state of `__x` inserted or in an error state.

5.979.5.2 `template<typename _UIntType, size_t __w, size_t __s, size_t __r> bool operator==(const subtract_with_carry_engine<_UIntType, __w, __s, __r> & __lhs, const subtract_with_carry_engine<_UIntType, __w, __s, __r> & __rhs) [friend]`

Compares two % `subtract_with_carry_engine` random number generator objects of the same type for equality.

Parameters

<code>__lhs</code>	A % subtract_with_carry_engine random number generator object.
<code>__rhs</code>	Another % subtract_with_carry_engine random number generator object.

Returns

true if the infinite sequences of generated values would be equal, false otherwise.

Definition at line 767 of file random.h.

```
5.979.5.3 template<typename _UIntType, size_t __w, size_t __s, size_t __r> template<typename _UIntType1, size_t __w1, size_t
    __s1, size_t __r1, typename _CharT, typename _Traits > std::basic_istream<_CharT, _Traits>& operator>> (
    std::basic_istream<_CharT, _Traits> & __is, std::subtract_with_carry_engine<_UIntType1, __w1, __s1, __r1
    > & __x ) [friend]
```

Extracts the current state of a % subtract_with_carry_engine random number generator engine `__x` from the input stream `__is`.

Parameters

<code>__is</code>	An input stream.
<code>__x</code>	A % subtract_with_carry_engine random number generator engine.

Returns

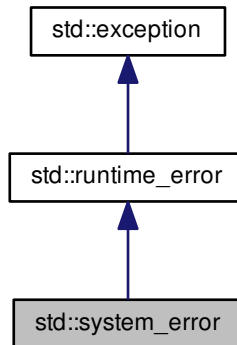
The input stream with the state of `__x` extracted or in an error state.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.980 std::system_error Class Reference

Inheritance diagram for std::system_error:



Public Member Functions

- **system_error** ([error_code](#) __ec=[error_code](#)())
- **system_error** ([error_code](#) __ec, const [string](#) &__what)
- **system_error** ([error_code](#) __ec, const char *__what)
- **system_error** (int __v, const error_category &__ecat, const char *__what)
- **system_error** (int __v, const error_category &__ecat)
- **system_error** (int __v, const error_category &__ecat, const [string](#) &__what)
- const [error_code](#) & **code** () const noexcept
- virtual const char * **what** () const _GLIBCXX_TXN_SAFE_DYN noexcept

5.980.1 Detailed Description

Thrown to indicate error code of underlying system.

Definition at line 333 of file system_error.

5.980.2 Member Function Documentation

5.980.2.1 virtual const char* std::runtime_error::what () const [virtual], [noexcept], [inherited]

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

The documentation for this class was generated from the following file:

- [system_error](#)

5.981 std::thread Class Reference

Classes

- class [id](#)

Public Types

- using **_State_ptr** = [unique_ptr](#)<_State >
- typedef __gthread_t **native_handle_type**

Public Member Functions

- **thread** ([thread](#) &)=delete
- **thread** (const [thread](#) &)=delete
- **thread** ([thread](#) &&__t) noexcept
- template<typename _Callable , typename... _Args>
thread (_Callable &&__f, _Args &&...__args)
- void **detach** ()
- [thread::id](#) **get_id** () const noexcept
- void **join** ()
- bool **joinable** () const noexcept
- native_handle_type [native_handle](#) ()
- [thread](#) & **operator=** (const [thread](#) &)=delete
- [thread](#) & **operator=** ([thread](#) &&__t) noexcept
- void **swap** ([thread](#) &__t) noexcept

Static Public Member Functions

- static unsigned int **hardware_concurrency** () noexcept

5.981.1 Detailed Description

[thread](#)

Definition at line 61 of file [thread](#).

5.981.2 Member Function Documentation

5.981.2.1 [native_handle_type](#) [std::thread::native_handle](#) () [\[inline\]](#)

Precondition

[thread](#) is joinable

Definition at line 179 of file [thread](#).

The documentation for this class was generated from the following file:

- [thread](#)

5.982 `std::thread::id` Class Reference

Public Member Functions

- `id` (`native_handle_type __id`)

Friends

- class `hash< thread::id >`
- bool `operator< (thread::id __x, thread::id __y)` noexcept
- template<class `_CharT`, class `_Traits` >
`basic_ostream< _CharT, _Traits > & operator<< (basic_ostream< _CharT, _Traits > &__out, thread::id __id)`
- bool `operator== (thread::id __x, thread::id __y)` noexcept
- class `thread`

5.982.1 Detailed Description

`thread::id`

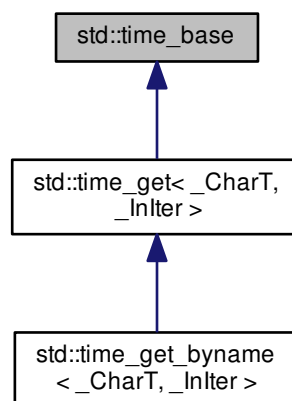
Definition at line 76 of file `thread`.

The documentation for this class was generated from the following file:

- [thread](#)

5.983 `std::time_base` Class Reference

Inheritance diagram for `std::time_base`:



Public Types

- enum **dateorder** {
 no_order, **dmy**, **mdy**, **ymd**,
 ydm }

5.983.1 Detailed Description

Time format ordering data.

This class provides an enum representing different orderings of time: day, month, and year.

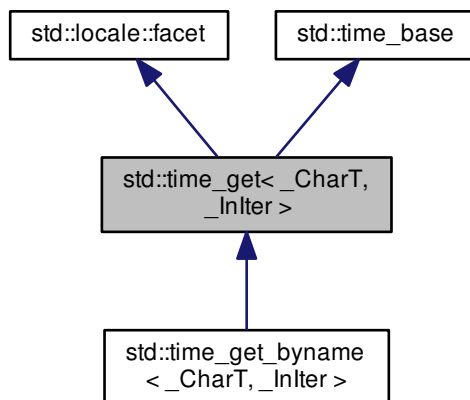
Definition at line 52 of file locale_facets_nonio.h.

The documentation for this class was generated from the following file:

- [locale_facets_nonio.h](#)

5.984 std::time_get<_CharT, _InIter> Class Template Reference

Inheritance diagram for std::time_get<_CharT, _InIter>:



Public Types

- enum **dateorder** {
 no_order, **dmy**, **mdy**, **ymd**,
 ydm }
- typedef `_CharT` [char_type](#)
- typedef `_InIter` [iter_type](#)

Public Member Functions

- `time_get` (size_t __refs=0)
- `dateorder` `date_order` () const
- `iter_type get` (iter_type __s, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm, char __format, char __modifier=0) const
- `iter_type get` (iter_type __s, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm, const char_type * __fmt, const char_type * __fmtend) const
- `iter_type get_date` (iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm) const
- `iter_type get_monthname` (iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm) const
- `iter_type get_time` (iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm) const
- `iter_type get_weekday` (iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm) const
- `iter_type get_year` (iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm) const

Static Public Attributes

- static `locale::id` id

Protected Member Functions

- virtual `~time_get` ()
- `iter_type M_extract_name` (iter_type __beg, iter_type __end, int & __member, const _CharT ** __names, size_t __indexlen, ios_base & __io, ios_base::iostate & __err) const
- `iter_type M_extract_num` (iter_type __beg, iter_type __end, int & __member, int __min, int __max, size_t __len, ios_base & __io, ios_base::iostate & __err) const
- `iter_type M_extract_via_format` (iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm, const _CharT * __format) const
- `iter_type M_extract_wday_or_month` (iter_type __beg, iter_type __end, int & __member, const _CharT ** __names, size_t __indexlen, ios_base & __io, ios_base::iostate & __err) const
- virtual `dateorder` `do_date_order` () const
- `iter_type do_get` (iter_type __s, iter_type __end, ios_base & __f, ios_base::iostate & __err, tm * __tm, char __format, char __modifier) const
- virtual `iter_type do_get_date` (iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm) const
- virtual `iter_type do_get_monthname` (iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm) const
- virtual `iter_type do_get_time` (iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm) const
- virtual `iter_type do_get_weekday` (iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm) const
- virtual `iter_type do_get_year` (iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm) const

Static Protected Member Functions

- static `_c_locale _S_clone_c_locale` (_c_locale & __cloc) throw ()
- static void `_S_create_c_locale` (_c_locale & __cloc, const char * __s, _c_locale __old=0)
- static void `_S_destroy_c_locale` (_c_locale & __cloc)
- static `_c_locale _S_get_c_locale` ()
- static const char * `_S_get_c_name` () throw ()
- static `_c_locale _S_lc_ctype_c_locale` (_c_locale __cloc, const char * __s)

5.984.1 Detailed Description

```
template<typename _CharT, typename _InIter>
class std::time_get<_CharT, _InIter>
```

Primary class template time_get.

This facet encapsulates the code to parse and return a date or time from a string. It is used by the istream numeric extraction operators.

The time_get template uses protected virtual functions to provide the actual results. The public accessors forward the call to the virtual functions. These virtual functions are hooks for developers to implement the behavior they require from the time_get facet.

Definition at line 366 of file locale_facets_nonio.h.

5.984.2 Member Typedef Documentation

5.984.2.1 `template<typename _CharT, typename _InIter> typedef _CharT std::time_get<_CharT, _InIter>::char_type`

Public typedefs.

Definition at line 372 of file locale_facets_nonio.h.

5.984.2.2 `template<typename _CharT, typename _InIter> typedef _InIter std::time_get<_CharT, _InIter>::iter_type`

Public typedefs.

Definition at line 373 of file locale_facets_nonio.h.

5.984.3 Constructor & Destructor Documentation

5.984.3.1 `template<typename _CharT, typename _InIter> std::time_get<_CharT, _InIter>::time_get (size_t __refs = 0)`
`[inline], [explicit]`

Constructor performs initialization.

This is the constructor provided by the standard.

Parameters

<code>__refs</code>	Passed to the base facet class.
---------------------	---------------------------------

Definition at line 387 of file locale_facets_nonio.h.

5.984.3.2 `template<typename _CharT, typename _Inlter > virtual std::time_get<_CharT, _Inlter >::~time_get ()`
`[inline], [protected], [virtual]`

Destructor.

Definition at line 591 of file `locale_facets_nonio.h`.

5.984.4 Member Function Documentation

5.984.4.1 `template<typename _CharT, typename _Inlter > dateorder std::time_get<_CharT, _Inlter >::date_order () const`
`[inline]`

Return preferred order of month, day, and year.

This function returns an enum from `time_base::dateorder` giving the preferred ordering if the format `x` given to `time_←put::put()` only uses month, day, and year. If the format `x` for the associated locale uses other fields, this function returns `time_base::dateorder::noorder`.

NOTE: The library always returns `noorder` at the moment.

Returns

A member of `time_base::dateorder`.

Definition at line 404 of file `locale_facets_nonio.h`.

5.984.4.2 `template<typename _CharT, typename _Inlter > _GLIBCXX_END_NAMESPACE_LDBL_OR_CXX11 time_base::dateorder`
`std::time_get<_CharT, _Inlter >::do_date_order () const` `[protected], [virtual]`

Return preferred order of month, day, and year.

This function returns an enum from `time_base::dateorder` giving the preferred ordering if the format `x` given to `time_←put::put()` only uses month, day, and year. This function is a hook for derived classes to change the value returned.

Returns

A member of `time_base::dateorder`.

Definition at line 626 of file `locale_facets_nonio.tcc`.

References `std::ios_base::M_getloc()`, `std::time_get<_CharT, _Inlter >::do_get_time()`, `std::ios_base::failbit`, `std::←ios_base::goodbit`, `std::min()`, `std::__ctype_abstract_base<_CharT >::narrow()`, and `std::__ctype_abstract_base<_CharT >::toupper()`.

5.984.4.3 `template<typename _CharT, typename _Inlter > _Inlter std::time_get<_CharT, _Inlter >::do_get (iter_type __s,`
`iter_type __end, ios_base & __f, ios_base::iostate & __err, tm * __tm, char __format, char __modifier) const`
`[inline], [protected]`

Parse input string according to format.

This function parses the string according to the provided format and optional modifier. This function is a hook for derived classes to change the value returned.

See also

`get()` for more details.

Parameters

<code>__s</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__f</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.
<code>__format</code>	Format specifier.
<code>__modifier</code>	Format modifier.

Returns

Iterator to first char not parsed.

Definition at line 1224 of file locale_facets_nonio.tcc.

References `std::ios_base::_M_getloc()`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::time_put< _CharT, _OutIter >::put()`, and `std::__ctype_abstract_base< _CharT >::widen()`.

Referenced by `std::time_get< _CharT, _InIter >::get()`.

5.984.4.4 `template<typename _CharT, typename _InIter> _InIter std::time_get< _CharT, _InIter >::do_get_date (iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm) const` `[protected]`, `[virtual]`

Parse input date string.

This function parses a date according to the format *X* and puts the results into a user-supplied struct tm. This function is a hook for derived classes to change the value returned.

See also

`get_date()` for details.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond date string.

Definition at line 1056 of file locale_facets_nonio.tcc.

References `std::ios_base::_M_getloc()`, `std::time_get<_CharT, _InIter>::do_get_weekday()`, and `std::ios_base::eofbit`.

Referenced by `std::time_get<_CharT, _InIter>::do_get_time()`.

5.984.4.5 `template<typename _CharT, typename _InIter> _InIter std::time_get<_CharT, _InIter>::do_get_monthname
(iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm) const
[protected], [virtual]`

Parse input month string.

This function parses a month name and puts the results into a user-supplied struct `tm`. This function is a hook for derived classes to change the value returned.

See also

`get_monthname()` for details.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct <code>tm</code> to fill in.

Returns

Iterator to first char beyond month name.

Definition at line 1100 of file `locale_facets_nonio.tcc`.

References `std::ios_base::_M_getloc()`, `std::time_get<_CharT, _InIter>::do_get_year()`, `std::ios_base::eofbit`, `std::ios_base::failbit`, and `std::ios_base::goodbit`.

Referenced by `std::time_get<_CharT, _InIter>::do_get_weekday()`.

5.984.4.6 `template<typename _CharT, typename _InIter> _InIter std::time_get<_CharT, _InIter>::do_get_time (iter_type
__beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm) const [protected],
[virtual]`

Parse input time string.

This function parses a time according to the format `x` and puts the results into a user-supplied struct `tm`. This function is a hook for derived classes to change the value returned.

See also

`get_time()` for details.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond time string.

Definition at line 1039 of file locale_facets_nonio.tcc.

References `std::ios_base::_M_getloc()`, `std::time_get< _CharT, _InIter >::do_get_date()`, and `std::ios_base::eofbit`.

Referenced by `std::time_get< _CharT, _InIter >::do_date_order()`.

```
5.984.4.7 template<typename _CharT, typename _InIter> _InIter std::time_get< _CharT, _InIter >::do_get_weekday
( iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm ) const
[protected], [virtual]
```

Parse input weekday string.

This function parses a weekday name and puts the results into a user-supplied struct tm. This function is a hook for derived classes to change the value returned.

See also

`get_weekday()` for details.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond weekday name.

Definition at line 1073 of file locale_facets_nonio.tcc.

References `std::ios_base::_M_getloc()`, `std::time_get< _CharT, _InIter >::do_get_monthname()`, `std::ios_base::eofbit`, `std::ios_base::failbit`, and `std::ios_base::goodbit`.

Referenced by `std::time_get< _CharT, _InIter >::do_get_date()`.

```
5.984.4.8 template<typename _CharT, typename _InIter > _InIter std::time_get< _CharT, _InIter >::do_get_year( iter_type
__beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm ) const [protected],
[virtual]
```

Parse input year string.

This function reads up to 4 characters to parse a year string and puts the results into a user-supplied struct tm. This function is a hook for derived classes to change the value returned.

See also

`get_year()` for details.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond year.

Definition at line 1127 of file `locale_facets_nonio.tcc`.

References `std::ios_base::M_getloc()`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::time_get< _CharT, _InIter >::get()`, and `std::ios_base::goodbit`.

Referenced by `std::time_get< _CharT, _InIter >::do_get_monthname()`.

```
5.984.4.9 template<typename _CharT, typename _InIter > iter_type std::time_get< _CharT, _InIter >::get( iter_type __s,
iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm, char __format, char __modifier = 0 )
const [inline]
```

Parse input string according to format.

This function calls `time_get::do_get` with the provided parameters.

See also

`do_get()` and `get()`.

Parameters

<code>__s</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.
<code>__format</code>	Format specifier.
<code>__modifier</code>	Format modifier.

Returns

Iterator to first char not parsed.

Definition at line 557 of file locale_facets_nonio.h.

Referenced by std::time_get< _CharT, _InIter >::do_get_year().

5.984.4.10 `template<typename _CharT, typename _InIter > _InIter std::time_get< _CharT, _InIter >::get (iter_type __s, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm, const char_type * __fmt, const char_type * __fmtend) const [inline]`

Parse input string according to format.

This function parses the input string according to a provided format string. It does the inverse of time_put::put. The format string follows the format specified for strptime(3)/strptime(3). The actual parsing is done by time_get::do_get.

Parameters

<code>__s</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.
<code>__fmt</code>	Start of the format string.
<code>__fmtend</code>	End of the format string.

Returns

Iterator to first char not parsed.

Definition at line 1152 of file locale_facets_nonio.tcc.

References std::ios_base::M_getloc(), std::time_get< _CharT, _InIter >::do_get(), std::ios_base::eofbit, std::ios_base::failbit, std::ios_base::goodbit, std::__ctype_abstract_base< _CharT >::is(), std::__ctype_abstract_base< _CharT >::narrow(), std::__ctype_abstract_base< _CharT >::tolower(), and std::__ctype_abstract_base< _CharT >::toupper().

5.984.4.11 `template<typename _CharT, typename _InIter > iter_type std::time_get< _CharT, _InIter >::get_date (iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm) const [inline]`

Parse input date string.

This function parses a date according to the format *x* and puts the results into a user-supplied struct tm. The result is returned by calling time_get::do_get_date().

If there is a valid date string according to format *x*, *tm* will be filled in accordingly and the returned iterator will point to the first character beyond the date string. If an error occurs before the end, *err* |= ios_base::failbit. If parsing reads all the characters, *err* |= ios_base::eofbit.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond date string.

Definition at line 453 of file locale_facets_nonio.h.

```
5.984.4.12 template<typename _CharT, typename _InIter > iter_type std::time_get< _CharT, _InIter >::get_monthname (
    iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm ) const [inline]
```

Parse input month string.

This function parses a month name and puts the results into a user-supplied struct tm. The result is returned by calling `time_get::do_get_monthname()`.

Parsing starts by parsing an abbreviated month name. If a valid abbreviation is followed by a character that would lead to the full month name, parsing continues until the full name is found or an error occurs. Otherwise parsing finishes at the end of the abbreviated name.

If an error occurs before the end, `err |= ios_base::failbit`. If parsing reads all the characters, `err |= ios_base::eofbit`.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond month name.

Definition at line 510 of file locale_facets_nonio.h.

```
5.984.4.13 template<typename _CharT, typename _InIter > iter_type std::time_get< _CharT, _InIter >::get_time ( iter_type
    __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm ) const [inline]
```

Parse input time string.

This function parses a time according to the format *X* and puts the results into a user-supplied struct *tm*. The result is returned by calling `time_get::do_get_time()`.

If there is a valid time string according to format *X*, *tm* will be filled in accordingly and the returned iterator will point to the first character beyond the time string. If an error occurs before the end, `err != ios_base::failbit`. If parsing reads all the characters, `err != ios_base::eofbit`.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct <i>tm</i> to fill in.

Returns

Iterator to first char beyond time string.

Definition at line 428 of file `locale_facets_nonio.h`.

5.984.4.14 `template<typename _CharT, typename _InIter > iter_type std::time_get<_CharT, _InIter >::get_weekday (iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm) const [inline]`

Parse input weekday string.

This function parses a weekday name and puts the results into a user-supplied struct *tm*. The result is returned by calling `time_get::do_get_weekday()`.

Parsing starts by parsing an abbreviated weekday name. If a valid abbreviation is followed by a character that would lead to the full weekday name, parsing continues until the full name is found or an error occurs. Otherwise parsing finishes at the end of the abbreviated name.

If an error occurs before the end, `err != ios_base::failbit`. If parsing reads all the characters, `err != ios_base::eofbit`.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct <i>tm</i> to fill in.

Returns

Iterator to first char beyond weekday name.

Definition at line 481 of file `locale_facets_nonio.h`.

5.984.4.15 `template<typename _CharT, typename _Inlter > iter_type std::time_get<_CharT, _Inlter >::get_year (iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm) const [inline]`

Parse input year string.

This function reads up to 4 characters to parse a year string and puts the results into a user-supplied struct tm. The result is returned by calling `time_get::do_get_year()`.

4 consecutive digits are interpreted as a full year. If there are exactly 2 consecutive digits, the library interprets this as the number of years since 1900.

If an error occurs before the end, `err |= ios_base::failbit`. If parsing reads all the characters, `err |= ios_base::eofbit`.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond year.

Definition at line 536 of file `locale_facets_nonio.h`.

5.984.5 Member Data Documentation

5.984.5.1 `template<typename _CharT, typename _Inlter > locale::id std::time_get<_CharT, _Inlter >::id [static]`

Numpunct facet id.

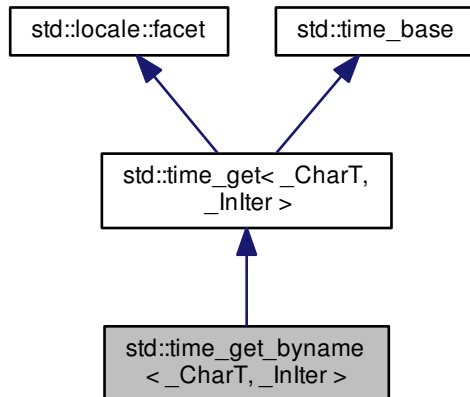
Definition at line 377 of file `locale_facets_nonio.h`.

The documentation for this class was generated from the following files:

- [locale_facets_nonio.h](#)
- [locale_facets_nonio.tcc](#)

5.985 std::time_get_byname<_CharT, _InIter> Class Template Reference

Inheritance diagram for std::time_get_byname<_CharT, _InIter>:



Public Types

- typedef `_CharT` **char_type**
- enum **dateorder** {
 no_order, **dmy**, **mdy**, **ymd**,
 ydm }
- typedef `_InIter` **iter_type**

Public Member Functions

- **time_get_byname** (const char *, size_t __refs=0)
- **time_get_byname** (const [string](#) &__s, size_t __refs=0)
- dateorder [date_order](#) () const
- **iter_type** get (iter_type __s, iter_type __end, ios_base &__io, ios_base::iostate &__err, tm *__tm, char __format, char __modifier=0) const
- **iter_type** get (iter_type __s, iter_type __end, ios_base &__io, ios_base::iostate &__err, tm *__tm, const [char_type](#) *__fmt, const [char_type](#) *__fmtend) const
- **iter_type** get_date (iter_type __beg, iter_type __end, ios_base &__io, ios_base::iostate &__err, tm *__tm) const
- **iter_type** get_monthname (iter_type __beg, iter_type __end, ios_base &__io, ios_base::iostate &__err, tm *__tm) const
- **iter_type** get_time (iter_type __beg, iter_type __end, ios_base &__io, ios_base::iostate &__err, tm *__tm) const
- **iter_type** get_weekday (iter_type __beg, iter_type __end, ios_base &__io, ios_base::iostate &__err, tm *__tm) const
- **iter_type** get_year (iter_type __beg, iter_type __end, ios_base &__io, ios_base::iostate &__err, tm *__tm) const

Static Public Attributes

- static `locale::id` `id`

Protected Member Functions

- `iter_type M_extract_name(iter_type __beg, iter_type __end, int &__member, const _CharT **__names, size_t __indexlen, ios_base &__io, ios_base::iostate &__err) const`
- `iter_type M_extract_num(iter_type __beg, iter_type __end, int &__member, int __min, int __max, size_t __len, ios_base &__io, ios_base::iostate &__err) const`
- `iter_type M_extract_via_format(iter_type __beg, iter_type __end, ios_base &__io, ios_base::iostate &__err, tm *__tm, const _CharT *__format) const`
- `iter_type M_extract_wday_or_month(iter_type __beg, iter_type __end, int &__member, const _CharT **__names, size_t __indexlen, ios_base &__io, ios_base::iostate &__err) const`
- virtual `dateorder do_date_order() const`
- `iter_type do_get(iter_type __s, iter_type __end, ios_base &__f, ios_base::iostate &__err, tm *__tm, char __format, char __modifier) const`
- virtual `iter_type do_get_date(iter_type __beg, iter_type __end, ios_base &__io, ios_base::iostate &__err, tm *__tm) const`
- virtual `iter_type do_get_monthname(iter_type __beg, iter_type __end, ios_base &__, ios_base::iostate &__err, tm *__tm) const`
- virtual `iter_type do_get_time(iter_type __beg, iter_type __end, ios_base &__io, ios_base::iostate &__err, tm *__tm) const`
- virtual `iter_type do_get_weekday(iter_type __beg, iter_type __end, ios_base &__, ios_base::iostate &__err, tm *__tm) const`
- virtual `iter_type do_get_year(iter_type __beg, iter_type __end, ios_base &__io, ios_base::iostate &__err, tm *__tm) const`

Static Protected Member Functions

- static `_c_locale S_clone_c_locale(_c_locale &__cloc) throw()`
- static `void S_create_c_locale(_c_locale &__cloc, const char *__s, _c_locale __old=0)`
- static `void S_destroy_c_locale(_c_locale &__cloc)`
- static `_c_locale S_get_c_locale()`
- static `const char * S_get_c_name() throw()`
- static `_c_locale S_lc_ctype_c_locale(_c_locale __cloc, const char *__s)`

5.985.1 Detailed Description

```
template<typename _CharT, typename _InIter>
class std::time_get_byname< _CharT, _InIter >
```

class `time_get_byname` [22.2.5.2].

Definition at line 758 of file `locale_facets_nonio.h`.

5.985.2 Member Function Documentation

5.985.2.1 `template<typename _CharT, typename _InIter> constexpr std::time_get<_CharT, _InIter>::date_order () const`
`[inline], [inherited]`

Return preferred order of month, day, and year.

This function returns an enum from `time_base::dateorder` giving the preferred ordering if the format `x` given to `time_←put::put()` only uses month, day, and year. If the format `x` for the associated locale uses other fields, this function returns `time_base::dateorder::noorder`.

NOTE: The library always returns `noorder` at the moment.

Returns

A member of `time_base::dateorder`.

Definition at line 404 of file `locale_facets_nonio.h`.

5.985.2.2 `template<typename _CharT, typename _InIter> _GLIBCXX_END_NAMESPACE_LDBL_OR_CXX11 time_base::dateorder`
`std::time_get<_CharT, _InIter>::do_date_order () const` `[protected], [virtual], [inherited]`

Return preferred order of month, day, and year.

This function returns an enum from `time_base::dateorder` giving the preferred ordering if the format `x` given to `time_←put::put()` only uses month, day, and year. This function is a hook for derived classes to change the value returned.

Returns

A member of `time_base::dateorder`.

Definition at line 626 of file `locale_facets_nonio.tcc`.

References `std::ios_base::M_getloc()`, `std::time_get<_CharT, _InIter>::do_get_time()`, `std::ios_base::failbit`, `std::←ios_base::goodbit`, `std::min()`, `std::__ctype_abstract_base<_CharT>::narrow()`, and `std::__ctype_abstract_base<_CharT>::toupper()`.

5.985.2.3 `template<typename _CharT, typename _InIter> _InIter std::time_get<_CharT, _InIter>::do_get (iter_type __s,`
`iter_type __end, ios_base & __f, ios_base::iostate & __err, tm * __tm, char __format, char __modifier) const`
`[inline], [protected], [inherited]`

Parse input string according to format.

This function parses the string according to the provided format and optional modifier. This function is a hook for derived classes to change the value returned.

See also

`get()` for more details.

Parameters

<code>__s</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__f</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.
<code>__format</code>	Format specifier.
<code>__modifier</code>	Format modifier.

Returns

Iterator to first char not parsed.

Definition at line 1224 of file locale_facets_nonio.tcc.

References `std::ios_base::_M_getloc()`, `std::ios_base::eofbit`, `std::ios_base::goodbit`, `std::time_put<_CharT, _OutIter>::put()`, and `std::__ctype_abstract_base<_CharT>::widen()`.

Referenced by `std::time_get<_CharT, _InIter>::get()`.

5.985.2.4 `template<typename _CharT, typename _InIter> _InIter std::time_get<_CharT, _InIter>::do_get_date (iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm) const` `[protected]`, `[virtual]`, `[inherited]`

Parse input date string.

This function parses a date according to the format *X* and puts the results into a user-supplied struct tm. This function is a hook for derived classes to change the value returned.

See also

`get_date()` for details.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond date string.

Definition at line 1056 of file locale_facets_nonio.tcc.

References std::ios_base::_M_getloc(), std::time_get< _CharT, _InIter >::do_get_weekday(), and std::ios_base::eofbit.

Referenced by std::time_get< _CharT, _InIter >::do_get_time().

5.985.2.5 `template<typename _CharT, typename _InIter > _InIter std::time_get< _CharT, _InIter >::do_get_monthname
(iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm) const
[protected], [virtual], [inherited]`

Parse input month string.

This function parses a month name and puts the results into a user-supplied struct tm. This function is a hook for derived classes to change the value returned.

See also

get_monthname() for details.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond month name.

Definition at line 1100 of file locale_facets_nonio.tcc.

References std::ios_base::_M_getloc(), std::time_get< _CharT, _InIter >::do_get_year(), std::ios_base::eofbit, std::ios_base::failbit, and std::ios_base::goodbit.

Referenced by std::time_get< _CharT, _InIter >::do_get_weekday().

5.985.2.6 `template<typename _CharT, typename _InIter > _InIter std::time_get< _CharT, _InIter >::do_get_time (iter_type
__beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm) const [protected],
[virtual], [inherited]`

Parse input time string.

This function parses a time according to the format x and puts the results into a user-supplied struct tm. This function is a hook for derived classes to change the value returned.

See also

get_time() for details.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond time string.

Definition at line 1039 of file locale_facets_nonio.tcc.

References `std::ios_base::_M_getloc()`, `std::time_get<_CharT, _InIter>::do_get_date()`, and `std::ios_base::eofbit`.

Referenced by `std::time_get<_CharT, _InIter>::do_date_order()`.

```
5.985.2.7 template<typename _CharT, typename _InIter> _InIter std::time_get<_CharT, _InIter>::do_get_weekday
( iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm ) const
[protected], [virtual], [inherited]
```

Parse input weekday string.

This function parses a weekday name and puts the results into a user-supplied struct tm. This function is a hook for derived classes to change the value returned.

See also

`get_weekday()` for details.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond weekday name.

Definition at line 1073 of file locale_facets_nonio.tcc.

References `std::ios_base::_M_getloc()`, `std::time_get<_CharT, _InIter>::do_get_monthname()`, `std::ios_base::eofbit`, `std::ios_base::failbit`, and `std::ios_base::goodbit`.

Referenced by `std::time_get<_CharT, _InIter>::do_get_date()`.

5.985.2.8 `template<typename _CharT, typename _InIter> _InIter std::time_get<_CharT, _InIter>::do_get_year(iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm) const` [protected], [virtual], [inherited]

Parse input year string.

This function reads up to 4 characters to parse a year string and puts the results into a user-supplied struct tm. This function is a hook for derived classes to change the value returned.

See also

`get_year()` for details.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond year.

Definition at line 1127 of file `locale_facets_nonio.tcc`.

References `std::ios_base::M_getloc()`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::time_get<_CharT, _InIter>::get()`, and `std::ios_base::goodbit`.

Referenced by `std::time_get<_CharT, _InIter>::do_get_monthname()`.

5.985.2.9 `template<typename _CharT, typename _InIter> iter_type std::time_get<_CharT, _InIter>::get(iter_type __s, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm, char __format, char __modifier = 0) const` [inline], [inherited]

Parse input string according to format.

This function calls `time_get::do_get` with the provided parameters.

See also

`do_get()` and `get()`.

Parameters

<code>__s</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.
<code>__format</code>	Format specifier.
<code>__modifier</code>	Format modifier.

Returns

Iterator to first char not parsed.

Definition at line 557 of file locale_facets_nonio.h.

Referenced by `std::time_get<_CharT, _InIter>::do_get_year()`.

5.985.2.10 `template<typename _CharT, typename _InIter> _InIter std::time_get<_CharT, _InIter>::get (iter_type __s, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm, const char_type * __fmt, const char_type * __fmtend) const [inline], [inherited]`

Parse input string according to format.

This function parses the input string according to a provided format string. It does the inverse of `time_put::put`. The format string follows the format specified for `strptime(3)/strptime(3)`. The actual parsing is done by `time_get::do_get`.

Parameters

<code>__s</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.
<code>__fmt</code>	Start of the format string.
<code>__fmtend</code>	End of the format string.

Returns

Iterator to first char not parsed.

Definition at line 1152 of file locale_facets_nonio.tcc.

References `std::ios_base::M_getloc()`, `std::time_get<_CharT, _InIter>::do_get()`, `std::ios_base::eofbit`, `std::ios_base::failbit`, `std::ios_base::goodbit`, `std::__ctype_abstract_base<_CharT>::is()`, `std::__ctype_abstract_base<_CharT>::narrow()`, `std::__ctype_abstract_base<_CharT>::tolower()`, and `std::__ctype_abstract_base<_CharT>::toupper()`.

5.985.2.11 `template<typename _CharT, typename _InIter> iter_type std::time_get<_CharT, _InIter>::get_date (iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm) const [inline], [inherited]`

Parse input date string.

This function parses a date according to the format `x` and puts the results into a user-supplied struct tm. The result is returned by calling `time_get::do_get_date()`.

If there is a valid date string according to format `x`, `tm` will be filled in accordingly and the returned iterator will point to the first character beyond the date string. If an error occurs before the end, `err` |= `ios_base::failbit`. If parsing reads all the characters, `err` |= `ios_base::eofbit`.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond date string.

Definition at line 453 of file locale_facets_nonio.h.

```
5.985.2.12 template<typename _CharT, typename _InIter > iter_type std::time_get<_CharT, _InIter >::get_monthname
( iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm ) const
    [inline], [inherited]
```

Parse input month string.

This function parses a month name and puts the results into a user-supplied struct tm. The result is returned by calling `time_get::do_get_monthname()`.

Parsing starts by parsing an abbreviated month name. If a valid abbreviation is followed by a character that would lead to the full month name, parsing continues until the full name is found or an error occurs. Otherwise parsing finishes at the end of the abbreviated name.

If an error occurs before the end, `err |= ios_base::failbit`. If parsing reads all the characters, `err |= ios_base::eofbit`.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond month name.

Definition at line 510 of file locale_facets_nonio.h.

```
5.985.2.13 template<typename _CharT, typename _InIter > iter_type std::time_get<_CharT, _InIter >::get_time ( iter_type
__beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm ) const    [inline],
[inherited]
```

Parse input time string.

This function parses a time according to the format *X* and puts the results into a user-supplied struct *tm*. The result is returned by calling `time_get::do_get_time()`.

If there is a valid time string according to format *X*, *tm* will be filled in accordingly and the returned iterator will point to the first character beyond the time string. If an error occurs before the end, `err != ios_base::failbit`. If parsing reads all the characters, `err != ios_base::eofbit`.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct <i>tm</i> to fill in.

Returns

Iterator to first char beyond time string.

Definition at line 428 of file `locale_facets_nonio.h`.

```
5.985.2.14 template<typename _CharT, typename _InIter > iter_type std::time_get< _CharT, _InIter >::get_weekday
( iter_type __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm ) const
[inline], [inherited]
```

Parse input weekday string.

This function parses a weekday name and puts the results into a user-supplied struct *tm*. The result is returned by calling `time_get::do_get_weekday()`.

Parsing starts by parsing an abbreviated weekday name. If a valid abbreviation is followed by a character that would lead to the full weekday name, parsing continues until the full name is found or an error occurs. Otherwise parsing finishes at the end of the abbreviated name.

If an error occurs before the end, `err != ios_base::failbit`. If parsing reads all the characters, `err != ios_base::eofbit`.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct <i>tm</i> to fill in.

Returns

Iterator to first char beyond weekday name.

Definition at line 481 of file `locale_facets_nonio.h`.

```
5.985.2.15 template<typename _CharT, typename _InIter> iter_type std::time_get<_CharT, _InIter>::get_year ( iter_type
    __beg, iter_type __end, ios_base & __io, ios_base::iostate & __err, tm * __tm ) const    [inline],
    [inherited]
```

Parse input year string.

This function reads up to 4 characters to parse a year string and puts the results into a user-supplied struct tm. The result is returned by calling time_get::do_get_year().

4 consecutive digits are interpreted as a full year. If there are exactly 2 consecutive digits, the library interprets this as the number of years since 1900.

If an error occurs before the end, err |= ios_base::failbit. If parsing reads all the characters, err |= ios_base::eofbit.

Parameters

<code>__beg</code>	Start of string to parse.
<code>__end</code>	End of string to parse.
<code>__io</code>	Source of the locale.
<code>__err</code>	Error flags to set.
<code>__tm</code>	Pointer to struct tm to fill in.

Returns

Iterator to first char beyond year.

Definition at line 536 of file locale_facets_nonio.h.

5.985.3 Member Data Documentation

```
5.985.3.1 template<typename _CharT, typename _InIter> locale::id std::time_get<_CharT, _InIter>::id    [static],
    [inherited]
```

Numpunct facet id.

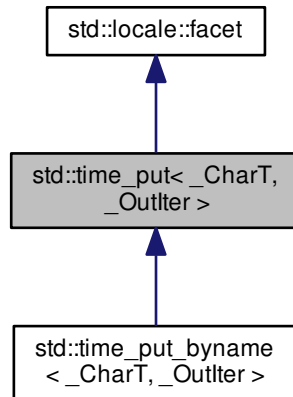
Definition at line 377 of file locale_facets_nonio.h.

The documentation for this class was generated from the following file:

- [locale_facets_nonio.h](#)

5.986 std::time_put<_CharT, _Outlter> Class Template Reference

Inheritance diagram for std::time_put<_CharT, _Outlter>:



Public Types

- typedef `_CharT` `char_type`
- typedef `_Outlter` `iter_type`

Public Member Functions

- `time_put` (`size_t` __refs=0)
- `iter_type put` (`iter_type` __s, `ios_base` &__io, `char_type` __fill, const `tm` *__tm, const `_CharT` *__beg, const `_CharT` *__end) const
- `iter_type put` (`iter_type` __s, `ios_base` &__io, `char_type` __fill, const `tm` *__tm, char __format, char __mod=0) const

Static Public Attributes

- static `locale::id` id

Protected Member Functions

- virtual `~time_put` ()
- virtual `iter_type do_put` (`iter_type` __s, `ios_base` &__io, `char_type` __fill, const `tm` *__tm, char __format, char __mod) const

Static Protected Member Functions

- static __c_locale **_S_clone_c_locale** (__c_locale &__cloc) throw ()
- static void **_S_create_c_locale** (__c_locale &__cloc, const char *__s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale &__cloc)
- static __c_locale **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static __c_locale **_S_lc_ctype_c_locale** (__c_locale __cloc, const char *__s)

5.986.1 Detailed Description

```
template<typename _CharT, typename _Outiter>
class std::time_put<_CharT, _Outiter>
```

Primary class template time_put.

This facet encapsulates the code to format and output dates and times according to formats used by strftime().

The time_put template uses protected virtual functions to provide the actual results. The public accessors forward the call to the virtual functions. These virtual functions are hooks for developers to implement the behavior they require from the time_put facet.

Definition at line 795 of file locale_facets_nonio.h.

5.986.2 Member Typedef Documentation

5.986.2.1 `template<typename _CharT, typename _Outiter> typedef _CharT std::time_put<_CharT, _Outiter>::char_type`

Public typedefs.

Definition at line 801 of file locale_facets_nonio.h.

5.986.2.2 `template<typename _CharT, typename _Outiter> typedef _Outiter std::time_put<_CharT, _Outiter>::iter_type`

Public typedefs.

Definition at line 802 of file locale_facets_nonio.h.

5.986.3 Constructor & Destructor Documentation

5.986.3.1 `template<typename _CharT, typename _Outiter> std::time_put<_CharT, _Outiter>::time_put (size_t __refs = 0)`
`[inline], [explicit]`

Constructor performs initialization.

This is the constructor provided by the standard.

Parameters

<code>__refs</code>	Passed to the base facet class.
---------------------	---------------------------------

Definition at line 816 of file `locale_facets_nonio.h`.

5.986.3.2 `template<typename _CharT, typename _Outlter > virtual std::time_put< _CharT, _Outlter >::~time_put ()`
`[inline], [protected], [virtual]`

Destructor.

Definition at line 862 of file `locale_facets_nonio.h`.

5.986.4 Member Function Documentation

5.986.4.1 `template<typename _CharT, typename _Outlter > _Outlter std::time_put< _CharT, _Outlter >::do_put (iter_type`
`__s, ios_base & __io, char_type __fill, const tm * __tm, char __format, char __mod) const` `[protected],`
`[virtual]`

Format and output a time or date.

This function formats the data in struct `tm` according to the provided format char and optional modifier. This function is a hook for derived classes to change the value returned.

See also

`put()` for more details.

Parameters

<code>__s</code>	The stream to write to.
<code>__io</code>	Source of locale.
<code>__fill</code>	<code>char_type</code> to use for padding.
<code>__tm</code>	Struct <code>tm</code> with date and time info to format.
<code>__format</code>	Format char.
<code>__mod</code>	Optional modifier char.

Returns

Iterator after writing.

Definition at line 1292 of file `locale_facets_nonio.tcc`.

References `std::ios_base::M_getloc()`, and `std::__ctype_abstract_base< _CharT >::widen()`.

Referenced by `std::time_put< _CharT, _Outlter >::put()`.

5.986.4.2 `template<typename _CharT, typename _OutIter > _OutIter std::time_put< _CharT, _OutIter >::put (iter_type __s, ios_base & __io, char_type __fill, const tm * __tm, const _CharT * __beg, const _CharT * __end) const`

Format and output a time or date.

This function formats the data in struct tm according to the provided format string. The format string is interpreted as by `strftime()`.

Parameters

<code>__s</code>	The stream to write to.
<code>__io</code>	Source of locale.
<code>__fill</code>	char_type to use for padding.
<code>__tm</code>	Struct tm with date and time info to format.
<code>__beg</code>	Start of format string.
<code>__end</code>	End of format string.

Returns

Iterator after writing.

Definition at line 1257 of file `locale_facets_nonio.tcc`.

References `std::ios_base::M_getloc()`, `std::time_put< _CharT, _OutIter >::do_put()`, and `std::__ctype_abstract_base< _CharT >::narrow()`.

Referenced by `std::time_get< _CharT, _InIter >::do_get()`.

5.986.4.3 `template<typename _CharT, typename _OutIter > iter_type std::time_put< _CharT, _OutIter >::put (iter_type __s, ios_base & __io, char_type __fill, const tm * __tm, char __format, char __mod = 0) const [inline]`

Format and output a time or date.

This function formats the data in struct tm according to the provided format char and optional modifier. The format and modifier are interpreted as by `strftime()`. It does so by returning `time_put::do_put()`.

Parameters

<code>__s</code>	The stream to write to.
<code>__io</code>	Source of locale.
<code>__fill</code>	char_type to use for padding.
<code>__tm</code>	Struct tm with date and time info to format.
<code>__format</code>	Format char.
<code>__mod</code>	Optional modifier char.

Returns

Iterator after writing.

Definition at line 855 of file locale_facets_nonio.h.

5.986.5 Member Data Documentation

5.986.5.1 `template<typename _CharT, typename _OutIter> locale::id std::time_put<_CharT, _OutIter>::id` `[static]`

Numpunct facet id.

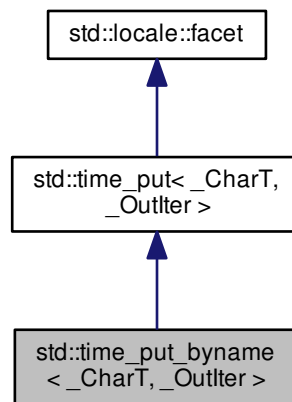
Definition at line 806 of file locale_facets_nonio.h.

The documentation for this class was generated from the following files:

- [locale_facets_nonio.h](#)
- [locale_facets_nonio.tcc](#)

5.987 std::time_put_byname<_CharT, _OutIter> Class Template Reference

Inheritance diagram for `std::time_put_byname<_CharT, _OutIter>`:

**Public Types**

- typedef `_CharT` **char_type**
- typedef `_OutIter` **iter_type**

Public Member Functions

- **time_put_byname** (const char *, size_t __refs=0)
- **time_put_byname** (const string & __s, size_t __refs=0)
- **iter_type put** (iter_type __s, ios_base & __io, char_type __fill, const tm * __tm, const _CharT * __beg, const _CharT * __end) const
- **iter_type put** (iter_type __s, ios_base & __io, char_type __fill, const tm * __tm, char __format, char __mod=0) const

Static Public Attributes

- static locale::id id

Protected Member Functions

- virtual **iter_type do_put** (iter_type __s, ios_base & __io, char_type __fill, const tm * __tm, char __format, char __mod) const

Static Protected Member Functions

- static __c_locale **_S_clone_c_locale** (__c_locale & __cloc) throw ()
- static void **_S_create_c_locale** (__c_locale & __cloc, const char * __s, __c_locale __old=0)
- static void **_S_destroy_c_locale** (__c_locale & __cloc)
- static __c_locale **_S_get_c_locale** ()
- static const char * **_S_get_c_name** () throw ()
- static __c_locale **_S_lc_ctype_c_locale** (__c_locale __cloc, const char * __s)

5.987.1 Detailed Description

```
template<typename _CharT, typename _Outlter>
class std::time_put_byname<_CharT, _Outlter>
```

class time_put_byname [22.2.5.4].

Definition at line 891 of file locale_facets_nonio.h.

5.987.2 Member Function Documentation

5.987.2.1 **template<typename _CharT, typename _Outlter> _Outlter std::time_put<_CharT, _Outlter>::do_put (iter_type __s, ios_base & __io, char_type __fill, const tm * __tm, char __format, char __mod) const** [protected], [virtual], [inherited]

Format and output a time or date.

This function formats the data in struct tm according to the provided format char and optional modifier. This function is a hook for derived classes to change the value returned.

See also

put() for more details.

Parameters

<code>__s</code>	The stream to write to.
<code>__io</code>	Source of locale.
<code>__fill</code>	<code>char_type</code> to use for padding.
<code>__tm</code>	Struct <code>tm</code> with date and time info to format.
<code>__format</code>	Format char.
<code>__mod</code>	Optional modifier char.

Returns

Iterator after writing.

Definition at line 1292 of file `locale_facets_nonio.tcc`.

References `std::ios_base::_M_getloc()`, and `std::__ctype_abstract_base<_CharT>::widen()`.

Referenced by `std::time_put<_CharT, _Outlter>::put()`.

5.987.2.2 `template<typename _CharT, typename _Outlter> _Outlter std::time_put<_CharT, _Outlter>::put (iter_type
__s, ios_base & __io, char_type __fill, const tm * __tm, const _CharT * __beg, const _CharT * __end) const
[inherited]`

Format and output a time or date.

This function formats the data in struct `tm` according to the provided format string. The format string is interpreted as by `strftime()`.

Parameters

<code>__s</code>	The stream to write to.
<code>__io</code>	Source of locale.
<code>__fill</code>	<code>char_type</code> to use for padding.
<code>__tm</code>	Struct <code>tm</code> with date and time info to format.
<code>__beg</code>	Start of format string.
<code>__end</code>	End of format string.

Returns

Iterator after writing.

Definition at line 1257 of file `locale_facets_nonio.tcc`.

References `std::ios_base::_M_getloc()`, `std::time_put<_CharT, _Outlter>::do_put()`, and `std::__ctype_abstract_base<_CharT>::narrow()`.

Referenced by `std::time_get<_CharT, _Inlter>::do_get()`.

5.987.2.3 `template<typename _CharT, typename _Outiter > iter_type std::time_put<_CharT, _Outiter>::put (iter_type __s, ios_base & __io, char_type __fill, const tm * __tm, char __format, char __mod = 0) const` `[inline]`, `[inherited]`

Format and output a time or date.

This function formats the data in struct tm according to the provided format char and optional modifier. The format and modifier are interpreted as by strftime(). It does so by returning time_put::do_put().

Parameters

<code>__s</code>	The stream to write to.
<code>__io</code>	Source of locale.
<code>__fill</code>	char_type to use for padding.
<code>__tm</code>	Struct tm with date and time info to format.
<code>__format</code>	Format char.
<code>__mod</code>	Optional modifier char.

Returns

Iterator after writing.

Definition at line 855 of file locale_facets_nonio.h.

5.987.3 Member Data Documentation

5.987.3.1 `template<typename _CharT, typename _Outiter > locale::id std::time_put<_CharT, _Outiter>::id` `[static]`, `[inherited]`

Numpunct facet id.

Definition at line 806 of file locale_facets_nonio.h.

The documentation for this class was generated from the following file:

- [locale_facets_nonio.h](#)

5.988 std::timed_mutex Class Reference

Public Member Functions

- **timed_mutex** (const [timed_mutex](#) &)=delete
- void **lock** ()
- [timed_mutex](#) & **operator=** (const [timed_mutex](#) &)=delete
- bool **try_lock** ()
- template<typename _Rep, typename _Period >
bool **try_lock_for** (const [chrono::duration](#)< _Rep, _Period > &__rtime)
- template<typename _Clock, typename _Duration >
bool **try_lock_until** (const [chrono::time_point](#)< _Clock, _Duration > &__atime)
- void **unlock** ()

5.988.1 Detailed Description

timed_mutex

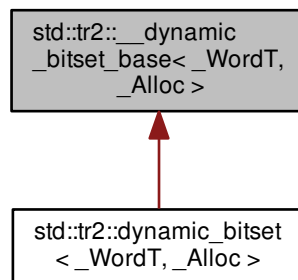
Definition at line 296 of file mutex.

The documentation for this class was generated from the following file:

- [mutex](#)

5.989 std::tr2::__dynamic_bitset_base< _WordT, _Alloc > Struct Template Reference

Inheritance diagram for std::tr2::__dynamic_bitset_base< _WordT, _Alloc >:



Public Types

- typedef `_Alloc` **allocator_type**
- typedef `_WordT` **block_type**
- typedef `size_t` **size_type**

Public Member Functions

- **__dynamic_bitset_base** (`const allocator_type &__alloc=allocator_type()`)
- **__dynamic_bitset_base** (`__dynamic_bitset_base &&__b`)
- **__dynamic_bitset_base** (`size_type __nbits, unsigned long long __val=0ULL, const allocator_type &__alloc=allocator_type()`)
- `size_t` **_M_are_all_aux** () const
- void **_M_assign** (`const __dynamic_bitset_base &__b`)
- void **_M_clear** ()
- void **_M_do_and** (`const __dynamic_bitset_base &__x`)
- void **_M_do_append_block** (`block_type __block, size_type __pos`)

- `size_t _M_do_count ()` const
- `void _M_do_dif (const __dynamic_bitset_base &__x)`
- `size_type _M_do_find_first (size_t __not_found)` const
- `size_type _M_do_find_next (size_t __prev, size_t __not_found)` const
- `void _M_do_flip ()`
- `void _M_do_left_shift (size_t __shift)`
- `void _M_do_or (const __dynamic_bitset_base &__x)`
- `void _M_do_reset ()`
- `void _M_do_right_shift (size_t __shift)`
- `void _M_do_set ()`
- `unsigned long long _M_do_to_ullong ()` const
- `unsigned long _M_do_to_ulong ()` const
- `void _M_do_xor (const __dynamic_bitset_base &__x)`
- `allocator_type _M_get_allocator ()` const
- `block_type & _M_getword (size_type __pos)`
- `block_type _M_getword (size_type __pos)` const
- `block_type & _M_hiword ()`
- `block_type _M_hiword ()` const
- `bool _M_is_any ()` const
- `bool _M_is_equal (const __dynamic_bitset_base &__x)` const
- `bool _M_is_less (const __dynamic_bitset_base &__x)` const
- `bool _M_is_proper_subset_of (const __dynamic_bitset_base &__b)` const
- `bool _M_is_subset_of (const __dynamic_bitset_base &__b)`
- `void _M_resize (size_t __nbits, bool __value)`
- `size_type _M_size ()` const noexcept
- `void _M_swap (__dynamic_bitset_base &__b)`

Static Public Member Functions

- `static block_type _S_maskbit (size_type __pos)` noexcept
- `static size_type _S_whichbit (size_type __pos)` noexcept
- `static size_type _S_whichbyte (size_type __pos)` noexcept
- `static size_type _S_whichword (size_type __pos)` noexcept

Public Attributes

- `std::vector< block_type, allocator_type > _M_w`

Static Public Attributes

- `static const size_type _S_bits_per_block`
- `static const size_type npos`

5.989.1 Detailed Description

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
struct std::tr2::__dynamic_bitset_base< _WordT, _Alloc >
```

Base class, general case.

See documentation for `dynamic_bitset`.

Definition at line 63 of file `dynamic_bitset`.

5.989.2 Member Data Documentation

```
5.989.2.1 template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
          std::vector<block_type, allocator_type> std::tr2::__dynamic_bitset_base< _WordT, _Alloc >::_M_w
```

0 is the least significant word.

Definition at line 76 of file `dynamic_bitset`.

The documentation for this struct was generated from the following files:

- [dynamic_bitset](#)
- [dynamic_bitset.tcc](#)

5.990 std::tr2::__reflection_typelist< _Elements > Struct Template Reference

5.990.1 Detailed Description

```
template<typename... _Elements>
struct std::tr2::__reflection_typelist< _Elements >
```

See N2965: Type traits and base classes by Michael Spertus Simple typelist. Compile-time list of types.

Definition at line 56 of file `tr2/type_traits`.

The documentation for this struct was generated from the following file:

- [tr2/type_traits](#)

5.991 std::tr2::__reflection_typelist< _First, _Rest... > Struct Template Reference

Public Types

- typedef [std::false_type](#) `empty`

5.991.1 Detailed Description

```
template<typename _First, typename... _Rest>
struct std::tr2::__reflection_typelist< _First, _Rest... >
```

Partial specialization.

Definition at line 67 of file `tr2/type_traits`.

The documentation for this struct was generated from the following file:

- [tr2/type_traits](#)

5.992 `std::tr2::__reflection_typelist<>` Struct Template Reference

Public Types

- typedef [std::true_type](#) `empty`

5.992.1 Detailed Description

```
template<>
struct std::tr2::__reflection_typelist<>
```

Specialization for an empty typelist.

Definition at line 60 of file `tr2/type_traits`.

The documentation for this struct was generated from the following file:

- [tr2/type_traits](#)

5.993 `std::tr2::bases<_Tp>` Struct Template Reference

Public Types

- typedef [__reflection_typelist<__bases\(_Tp\)...](#) `type`

5.993.1 Detailed Description

```
template<typename _Tp>
struct std::tr2::bases< _Tp >
```

Sequence abstraction metafunctions for manipulating a typelist.

Enumerate all the base classes of a class. Form of a typelist.

Definition at line 88 of file tr2/type_traits.

The documentation for this struct was generated from the following file:

- [tr2/type_traits](#)

5.994 std::tr2::bool_set Class Reference

Public Member Functions

- constexpr [bool_set](#) ()
- constexpr [bool_set](#) (bool __t)
- bool **contains** ([bool_set](#) __b) const
- bool **equals** ([bool_set](#) __b) const
- bool **is_emptyset** () const
- bool **is_indeterminate** () const
- bool **is_singleton** () const
- **operator bool** () const

Static Public Member Functions

- static [bool_set](#) **emptyset** ()
- static [bool_set](#) **indeterminate** ()

Friends

- [bool_set](#) **operator!** ([bool_set](#) __b)
- [bool_set](#) **operator&** ([bool_set](#) __s, [bool_set](#) __t)
- template<typename CharT , typename Traits >
[std::basic_ostream](#)< CharT, Traits > & **operator<<** ([std::basic_ostream](#)< CharT, Traits > &__out, [bool_set](#) __b)
- [bool_set](#) **operator==** ([bool_set](#) __s, [bool_set](#) __t)
- template<typename CharT , typename Traits >
[std::basic_istream](#)< CharT, Traits > & **operator>>** ([std::basic_istream](#)< CharT, Traits > &__in, [bool_set](#) &__b)
- [bool_set](#) **operator^** ([bool_set](#) __s, [bool_set](#) __t)
- [bool_set](#) **operator|** ([bool_set](#) __s, [bool_set](#) __t)

5.994.1 Detailed Description

bool_set

See N2136, Bool_set: multi-valued logic by Hervé Brönnimann, Guillaume Melquiond, Sylvain Pion.

The implicit conversion to bool is slippery! I may use the new explicit conversion. This has been specialized in the language so that in contexts requiring a bool the conversion happens implicitly. Thus most objections should be eliminated.

Definition at line 54 of file bool_set.

5.994.2 Constructor & Destructor Documentation

5.994.2.1 constexpr std::tr2::bool_set::bool_set () [inline]

Default constructor.

Definition at line 59 of file bool_set.

5.994.2.2 constexpr std::tr2::bool_set::bool_set (bool __t) [inline]

Constructor from bool.

Definition at line 62 of file bool_set.

5.994.3 Member Function Documentation

5.994.3.1 bool std::tr2::bool_set::equals (bool_set __b) const [inline]

Return true if states are equal.

Definition at line 69 of file bool_set.

5.994.3.2 bool std::tr2::bool_set::is_emptyset () const [inline]

Return true if this is empty.

Definition at line 73 of file bool_set.

5.994.3.3 bool std::tr2::bool_set::is_indeterminate () const [inline]

Return true if this is indeterminate.

Definition at line 77 of file bool_set.

5.994.3.4 `bool std::tr2::bool_set::is_singleton () const [inline]`

Return true if this is false or true (normal boolean).

Definition at line 81 of file `bool_set`.

5.994.3.5 `std::tr2::bool_set::operator bool () const [inline]`

Conversion to bool.

Definition at line 86 of file `bool_set`.

The documentation for this class was generated from the following files:

- [bool_set](#)
- [bool_set.tcc](#)

5.995 `std::tr2::direct_bases<_Tp>` Struct Template Reference

Public Types

- `typedef __reflection_typelist< __direct_bases(_Tp)... > type`

5.995.1 Detailed Description

```
template<typename _Tp>
struct std::tr2::direct_bases<_Tp>
```

Enumerate all the direct base classes of a class. Form of a typelist.

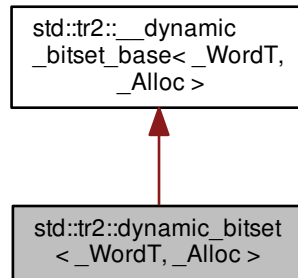
Definition at line 95 of file `tr2/type_traits`.

The documentation for this struct was generated from the following file:

- [tr2/type_traits](#)

5.996 std::tr2::dynamic_bitset< _WordT, _Alloc > Class Template Reference

Inheritance diagram for std::tr2::dynamic_bitset< _WordT, _Alloc >:



Classes

- class [reference](#)

Public Types

- typedef [__dynamic_bitset_base](#)< _WordT, _Alloc > **_Base**
- typedef _Alloc **allocator_type**
- typedef _WordT **block_type**
- typedef bool **const_reference**
- typedef size_t **size_type**

Public Member Functions

- [dynamic_bitset](#) (const allocator_type &__alloc=allocator_type())
- [dynamic_bitset](#) (size_type __nbits, unsigned long long __val=0ULL, const allocator_type &__alloc=allocator_type())
- **dynamic_bitset** ([initializer_list](#)< block_type > __il, const allocator_type &__alloc=allocator_type())
- template<typename _CharT, typename _Traits, typename _Alloc1 >
[dynamic_bitset](#) (const [std::basic_string](#)< _CharT, _Traits, _Alloc1 > &__str, typename [basic_string](#)< _CharT, _Traits, _Alloc1 >::size_type __pos=0, typename [basic_string](#)< _CharT, _Traits, _Alloc1 >::size_type __n=[std::basic_string](#)< _CharT, _Traits, _Alloc1 >::npos, _CharT __zero=_CharT('0'), _CharT __one=_CharT('1'), const allocator_type &__alloc=allocator_type())
- [dynamic_bitset](#) (const char *__str, const allocator_type &__alloc=allocator_type())
- [dynamic_bitset](#) (const [dynamic_bitset](#) &__b)
- [dynamic_bitset](#) ([dynamic_bitset](#) &&__b)
- template<typename _CharT, typename _Traits >
void **_M_copy_from_ptr** (const _CharT *, size_t, size_t, size_t, _CharT, _CharT)

- `template<typename _CharT, typename _Traits, typename _Alloc1 >`
`void _M_copy_from_string (const std::basic_string< _CharT, _Traits, _Alloc1 > &__str, size_t __pos, size_t __n, _CharT __zero=_CharT('0'), _CharT __one=_CharT('1'))`
- `template<typename _CharT, typename _Traits, typename _Alloc1 >`
`void _M_copy_to_string (std::basic_string< _CharT, _Traits, _Alloc1 > &__str, _CharT __zero=_CharT('0'), _CharT __one=_CharT('1')) const`
- `bool all () const`
- `bool any () const`
- `void append (block_type __block)`
- `void append (initializer_list< block_type > __il)`
- `template<typename _BlockInputIterator >`
`void append (_BlockInputIterator __first, _BlockInputIterator __last)`
- `void clear ()`
- `size_type count () const noexcept`
- `bool empty () const noexcept`
- `size_type find_first () const`
- `size_type find_next (size_t __prev) const`
- `dynamic_bitset< _WordT, _Alloc > & flip ()`
- `dynamic_bitset< _WordT, _Alloc > & flip (size_type __pos)`
- `allocator_type get_allocator () const`
- `bool is_proper_subset_of (const dynamic_bitset &__b) const`
- `bool is_subset_of (const dynamic_bitset &__b) const`
- `constexpr size_type max_size () noexcept`
- `bool none () const`
- `size_type num_blocks () const noexcept`
- `dynamic_bitset & operator= (const dynamic_bitset &__b)`
- `dynamic_bitset & operator= (dynamic_bitset && __b)`
- `dynamic_bitset< _WordT, _Alloc > operator~ () const`
- `void push_back (bool __bit)`
- `dynamic_bitset< _WordT, _Alloc > & reset ()`
- `dynamic_bitset< _WordT, _Alloc > & reset (size_type __pos)`
- `void resize (size_type __nbits, bool __value=false)`
- `dynamic_bitset< _WordT, _Alloc > & set ()`
- `dynamic_bitset< _WordT, _Alloc > & set (size_type __pos, bool __val=true)`
- `size_type size () const noexcept`
- `void swap (dynamic_bitset & __b)`
- `bool test (size_type __pos) const`
- `template<typename _CharT = char, typename _Traits = std::char_traits< _CharT>, typename _Alloc1 = std::allocator< _CharT>>`
`std::basic_string< _CharT, _Traits, _Alloc1 > to_string (_CharT __zero=_CharT('0'), _CharT __one=_CharT('1'))`
`const`
- `unsigned long long to_ullong () const`
- `unsigned long to_ulong () const`

- `dynamic_bitset< _WordT, _Alloc > & operator&= (const dynamic_bitset< _WordT, _Alloc > &__rhs)`
- `dynamic_bitset< _WordT, _Alloc > & operator&= (dynamic_bitset< _WordT, _Alloc > && __rhs)`
- `dynamic_bitset< _WordT, _Alloc > & operator|= (const dynamic_bitset< _WordT, _Alloc > &__rhs)`
- `dynamic_bitset< _WordT, _Alloc > & operator^= (const dynamic_bitset< _WordT, _Alloc > &__rhs)`
- `dynamic_bitset< _WordT, _Alloc > & operator-= (const dynamic_bitset< _WordT, _Alloc > &__rhs)`

- `dynamic_bitset< _WordT, _Alloc > & operator<<= (size_type __pos)`

- [dynamic_bitset<_WordT, _Alloc > & operator>>=](#) (size_type __pos)
- [reference operator\[\]](#) (size_type __pos)
- const_reference [operator\[\]](#) (size_type __pos) const
- [dynamic_bitset<_WordT, _Alloc > operator<<](#) (size_type __pos) const
- [dynamic_bitset<_WordT, _Alloc > operator>>](#) (size_type __pos) const

Static Public Attributes

- static const size_type **bits_per_block**
- static const size_type **npos**

Private Member Functions

- size_t **_M_are_all_aux** () const
- void **_M_assign** (const [__dynamic_bitset_base](#) &__b)
- void **_M_clear** ()
- void **_M_do_and** (const [__dynamic_bitset_base](#) &__x)
- void **_M_do_append_block** (block_type __block, size_type __pos)
- size_t **_M_do_count** () const
- void **_M_do_dif** (const [__dynamic_bitset_base](#) &__x)
- size_type **_M_do_find_first** (size_t __not_found) const
- size_type **_M_do_find_next** (size_t __prev, size_t __not_found) const
- void **_M_do_flip** ()
- void **_M_do_left_shift** (size_t __shift)
- void **_M_do_or** (const [__dynamic_bitset_base](#) &__x)
- void **_M_do_reset** ()
- void **_M_do_right_shift** (size_t __shift)
- void **_M_do_set** ()
- unsigned long long **_M_do_to_ullong** () const
- unsigned long **_M_do_to_ulong** () const
- void **_M_do_xor** (const [__dynamic_bitset_base](#) &__x)
- allocator_type **_M_get_allocator** () const
- block_type & **_M_getword** (size_type __pos)
- block_type **_M_getword** (size_type __pos) const
- block_type & **_M_hiword** ()
- block_type **_M_hiword** () const
- bool **_M_is_any** () const
- bool **_M_is_equal** (const [__dynamic_bitset_base](#) &__x) const
- bool **_M_is_less** (const [__dynamic_bitset_base](#) &__x) const
- bool **_M_is_proper_subset_of** (const [__dynamic_bitset_base](#) &__b) const
- bool **_M_is_subset_of** (const [__dynamic_bitset_base](#) &__b)
- void **_M_resize** (size_t __nbits, bool __value)
- size_type **_M_size** () const noexcept
- void **_M_swap** ([__dynamic_bitset_base](#) &__b)

Static Private Member Functions

- static `block_type _S_maskbit (size_type __pos) noexcept`
- static `size_type _S_whichbit (size_type __pos) noexcept`
- static `size_type _S_whichbyte (size_type __pos) noexcept`
- static `size_type _S_whichword (size_type __pos) noexcept`

Private Attributes

- `std::vector< block_type, allocator_type > _M_w`

Static Private Attributes

- static const `size_type _S_bits_per_block`

Friends

- bool `operator< (const dynamic_bitset< _WordT, _Alloc > &__lhs, const dynamic_bitset< _WordT, _Alloc > &__rhs)`
- bool `operator== (const dynamic_bitset< _WordT, _Alloc > &__lhs, const dynamic_bitset< _WordT, _Alloc > &__rhs)`
- class `reference`

5.996.1 Detailed Description

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
class std::tr2::dynamic_bitset< _WordT, _Alloc >
```

The `dynamic_bitset` class represents a sequence of bits.

See N2050, Proposal to Add a Dynamically Sizeable Bitset to the Standard Library.

In the general unoptimized case, storage is allocated in word-sized blocks. Let B be the number of bits in a word, then $(Nb+(B-1))/B$ words will be used for storage. $B - NbB$ bits are unused. (They are the high-order bits in the highest word.) It is a class invariant that those unused bits are always zero.

If you think of `dynamic_bitset` as "a simple array of bits," be aware that your mental picture is reversed: a `dynamic_bitset` behaves the same way as bits in integers do, with the bit at index 0 in the "least significant / right-hand" position, and the bit at index $Nb-1$ in the "most significant / left-hand" position. Thus, unlike other containers, a `dynamic_bitset`'s index "counts from right to left," to put it very loosely.

This behavior is preserved when translating to and from strings. For example, the first line of the following program probably prints "b('a') is 0001100001" on a modern ASCII system.

```

1 #include <dynamic_bitset>
2 #include <iostream>
3 #include <sstream>
4
5 using namespace std;
6
7 int main()
8 {
9     long          a = 'a';
10    dynamic_bitset<> b(a);
11
12    cout << "b('a') is " << b << endl;
13
14    ostringstream s;
15    s << b;
16    string str = s.str();
17    cout << "index 3 in the string is " << str[3] << " but\n"
18         << "index 3 in the bitset is " << b[3] << endl;
19 }

```

Most of the actual code isn't contained in `dynamic_bitset<>` itself, but in the base class `__dynamic_bitset_base`. The base class works with whole words, not with individual bits. This allows us to specialize `__dynamic_bitset_base` for the important special case where the `dynamic_bitset` is only a single word.

Extra confusion can result due to the fact that the storage for `__dynamic_bitset_base` is a vector, and is indexed as such. This is carefully encapsulated.

Definition at line 413 of file `dynamic_bitset`.

5.996.2 Constructor & Destructor Documentation

5.996.2.1 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
std::tr2::dynamic_bitset<_WordT, _Alloc>::dynamic_bitset (const allocator_type & __alloc =
allocator_type()) [inline], [explicit]`

All bits set to zero.

Definition at line 574 of file `dynamic_bitset`.

5.996.2.2 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
std::tr2::dynamic_bitset<_WordT, _Alloc>::dynamic_bitset (size_type __nbits, unsigned long long __val =
0ULL, const allocator_type & __alloc = allocator_type()) [inline], [explicit]`

Initial bits bitwise-copied from a single word (others set to zero).

Definition at line 580 of file `dynamic_bitset`.

5.996.2.3 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> template<typename
_CharT, typename _Traits, typename _Alloc1 > std::tr2::dynamic_bitset<_WordT, _Alloc>::dynamic_bitset
(const std::basic_string<_CharT, _Traits, _Alloc1> & __str, typename basic_string<_CharT, _Traits,
_Alloc1>::size_type __pos = 0, typename basic_string<_CharT, _Traits, _Alloc1>::size_type __n =
std::basic_string<_CharT, _Traits, _Alloc1>::npos, _CharT __zero = _CharT('0'),
_CharT __one = _CharT('1'), const allocator_type & __alloc = allocator_type()) [inline],
[explicit]`

Use a subset of a string.

Parameters

<code>__str</code>	A string of '0' and '1' characters.
<code>__pos</code>	Index of the first character in <code>__str</code> to use.
<code>__n</code>	The number of characters to copy.
<code>__zero</code>	The character to use for unset bits.
<code>__one</code>	The character to use for set bits.
<code>__alloc</code>	An allocator.

Exceptions

<code>std::out_of_range</code>	If <code>__pos</code> is bigger the size of <code>__str</code> .
<code>std::invalid_argument</code>	If a character appears in the string which is neither '0' nor '1'.

Definition at line 605 of file `dynamic_bitset`.

```
5.996.2.4  template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
            std::tr2::dynamic_bitset< _WordT, _Alloc >::dynamic_bitset ( const char * __str, const allocator_type & __alloc
            = allocator_type() ) [inline], [explicit]
```

Construct from a string.

Parameters

<code>__str</code>	A string of '0' and '1' characters.
<code>__alloc</code>	An allocator.

Exceptions

<code>std::invalid_argument</code>	If a character appears in the string which is neither '0' nor '1'.
------------------------------------	--

Definition at line 634 of file `dynamic_bitset`.

```
5.996.2.5  template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
            std::tr2::dynamic_bitset< _WordT, _Alloc >::dynamic_bitset ( const dynamic_bitset< _WordT, _Alloc > &
            __b ) [inline]
```

Copy constructor.

Definition at line 650 of file `dynamic_bitset`.

```
5.996.2.6  template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
            std::tr2::dynamic_bitset< _WordT, _Alloc >::dynamic_bitset ( dynamic_bitset< _WordT, _Alloc > && __b )
            [inline]
```

Move constructor.

Definition at line 657 of file `dynamic_bitset`.

5.996.3 Member Function Documentation

5.996.3.1 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> bool
std::tr2::dynamic_bitset<_WordT, _Alloc>::all () const [inline]`

Tests whether all the bits are on.

Returns

True if all the bits are set.

Definition at line 1046 of file `dynamic_bitset`.

5.996.3.2 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> bool
std::tr2::dynamic_bitset<_WordT, _Alloc>::any () const [inline]`

Tests whether any of the bits are on.

Returns

True if at least one bit is set.

Definition at line 1054 of file `dynamic_bitset`.

5.996.3.3 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> void
std::tr2::dynamic_bitset<_WordT, _Alloc>::append (block_type __block) [inline]`

Append a block.

Definition at line 740 of file `dynamic_bitset`.

5.996.3.4 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> template<typename
_BlockInputIterator > void std::tr2::dynamic_bitset<_WordT, _Alloc>::append (_BlockInputIterator __first,
_BlockInputIterator __last) [inline]`

Append an iterator range of blocks.

Definition at line 758 of file `dynamic_bitset`.

5.996.3.5 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> void
std::tr2::dynamic_bitset<_WordT, _Alloc>::clear () [inline]`

Clear the bitset.

Definition at line 718 of file `dynamic_bitset`.

5.996.3.6 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> size_type
std::tr2::dynamic_bitset<_WordT, _Alloc>::count () const [inline], [noexcept]`

Returns the number of bits which are set.

Definition at line 1002 of file `dynamic_bitset`.

5.996.3.7 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> bool
std::tr2::dynamic_bitset<_WordT, _Alloc>::empty () const [inline], [noexcept]`

Returns true if the `dynamic_bitset` is empty.

Definition at line 1017 of file `dynamic_bitset`.

5.996.3.8 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> size_type
std::tr2::dynamic_bitset<_WordT, _Alloc>::find_first () const [inline]`

Finds the index of the first "on" bit.

Returns

The index of the first bit set, or `size()` if not found.

See also

`find_next`

Definition at line 1082 of file `dynamic_bitset`.

5.996.3.9 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> size_type
std::tr2::dynamic_bitset<_WordT, _Alloc>::find_next (size_t __prev) const [inline]`

Finds the index of the next "on" bit after `prev`.

Returns

The index of the next bit set, or `size()` if not found.

Parameters

<code>__prev</code>	Where to start searching.
---------------------	---------------------------

See also

`find_first`

Definition at line 1092 of file `dynamic_bitset`.

```
5.996.3.10 template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset<_WordT, _Alloc>& std::tr2::dynamic_bitset<_WordT, _Alloc>::flip ( ) [inline]
```

Toggles every bit to its opposite value.

Definition at line 897 of file dynamic_bitset.

```
5.996.3.11 template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset<_WordT, _Alloc>& std::tr2::dynamic_bitset<_WordT, _Alloc>::flip ( size_type __pos )
[inline]
```

Toggles a given bit to its opposite value.

Parameters

<code>__pos</code>	The index of the bit.
--------------------	-----------------------

Exceptions

<code>std::out_of_range</code>	If <code>__pos</code> is bigger the size of the set.
--------------------------------	--

Definition at line 910 of file dynamic_bitset.

```
5.996.3.12 template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> allocator_type
std::tr2::dynamic_bitset<_WordT, _Alloc>::get_allocator ( ) const [inline]
```

Return the allocator for the bitset.

Definition at line 698 of file dynamic_bitset.

```
5.996.3.13 template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> constexpr size_type
std::tr2::dynamic_bitset<_WordT, _Alloc>::max_size ( ) [inline], [noexcept]
```

Returns the maximum size of a dynamic_bitset object having the same type as *this. The real answer is max() * bits_per_block but is likely to overflow.

Definition at line 1024 of file dynamic_bitset.

```
5.996.3.14 template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> bool
std::tr2::dynamic_bitset<_WordT, _Alloc>::none ( ) const [inline]
```

Tests whether any of the bits are on.

Returns

True if none of the bits are set.

Definition at line 1062 of file dynamic_bitset.

```
5.996.3.15  template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> size_type
            std::tr2::dynamic_bitset<_WordT, _Alloc >::num_blocks ( ) const  [inline], [noexcept]
```

Returns the total number of blocks.

Definition at line 1012 of file dynamic_bitset.

```
5.996.3.16  template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
            dynamic_bitset<_WordT, _Alloc>& std::tr2::dynamic_bitset<_WordT, _Alloc >::operator&= ( const
            dynamic_bitset<_WordT, _Alloc > &__rhs )  [inline]
```

Operations on dynamic_bitsets.

Parameters

<code>__rhs</code>	A same-sized dynamic_bitset.
--------------------	------------------------------

These should be self-explanatory.

Definition at line 773 of file dynamic_bitset.

```
5.996.3.17  template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
            dynamic_bitset<_WordT, _Alloc>& std::tr2::dynamic_bitset<_WordT, _Alloc >::operator&= (
            dynamic_bitset<_WordT, _Alloc > &&__rhs )  [inline]
```

Operations on dynamic_bitsets.

Parameters

<code>__rhs</code>	A same-sized dynamic_bitset.
--------------------	------------------------------

These should be self-explanatory.

Definition at line 780 of file dynamic_bitset.

```
5.996.3.18  template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
            dynamic_bitset<_WordT, _Alloc>& std::tr2::dynamic_bitset<_WordT, _Alloc >::operator-= ( const
            dynamic_bitset<_WordT, _Alloc > &__rhs )  [inline]
```

Operations on dynamic_bitsets.

Parameters

<code>__rhs</code>	A same-sized dynamic_bitset.
--------------------	------------------------------

These should be self-explanatory.

Definition at line 801 of file dynamic_bitset.

```
5.996.3.19 template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset<_WordT, _Alloc> std::tr2::dynamic_bitset<_WordT, _Alloc>::operator<< ( size_type __pos
) const [inline]
```

Self-explanatory.

Definition at line 1068 of file dynamic_bitset.

```
5.996.3.20 template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset<_WordT, _Alloc>& std::tr2::dynamic_bitset<_WordT, _Alloc>::operator<=< ( size_type
__pos ) [inline]
```

Operations on dynamic_bitsets.

Parameters

<code>__pos</code>	The number of places to shift.
--------------------	--------------------------------

These should be self-explanatory.

Definition at line 816 of file dynamic_bitset.

```
5.996.3.21 template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> dynamic_bitset&
std::tr2::dynamic_bitset<_WordT, _Alloc>::operator= ( const dynamic_bitset<_WordT, _Alloc> & __b )
[inline]
```

Assignment.

Definition at line 675 of file dynamic_bitset.

```
5.996.3.22 template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> dynamic_bitset&
std::tr2::dynamic_bitset<_WordT, _Alloc>::operator= ( dynamic_bitset<_WordT, _Alloc> && __b )
[inline]
```

Move assignment.

Definition at line 688 of file dynamic_bitset.

```
5.996.3.23 template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset<_WordT, _Alloc> std::tr2::dynamic_bitset<_WordT, _Alloc>::operator>> ( size_type __pos
) const [inline]
```

Self-explanatory.

Definition at line 1072 of file dynamic_bitset.

```
5.996.3.24 template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset<_WordT, _Alloc>& std::tr2::dynamic_bitset<_WordT, _Alloc>::operator>>= ( size_type
__pos ) [inline]
```

Operations on dynamic_bitsets.

Parameters

<code>__pos</code>	The number of places to shift.
--------------------	--------------------------------

These should be self-explanatory.

Definition at line 829 of file `dynamic_bitset`.

5.996.3.25 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> reference
std::tr2::dynamic_bitset<_WordT, _Alloc>::operator[] (size_type __pos) [inline]`

Array-indexing support.

Parameters

<code>__pos</code>	Index into the <code>dynamic_bitset</code> .
--------------------	--

Returns

A bool for a 'const `dynamic_bitset`'. For non-const bitsets, an instance of the reference proxy class.

Note

These operators do no range checking and throw no exceptions, as required by DR 11 to the standard.

Definition at line 932 of file `dynamic_bitset`.

5.996.3.26 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> const_reference
std::tr2::dynamic_bitset<_WordT, _Alloc>::operator[] (size_type __pos) const [inline]`

Array-indexing support.

Parameters

<code>__pos</code>	Index into the <code>dynamic_bitset</code> .
--------------------	--

Returns

A bool for a 'const `dynamic_bitset`'. For non-const bitsets, an instance of the reference proxy class.

Note

These operators do no range checking and throw no exceptions, as required by DR 11 to the standard.

Definition at line 936 of file `dynamic_bitset`.

```
5.996.3.27 template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset<_WordT, _Alloc>& std::tr2::dynamic_bitset<_WordT, _Alloc>::operator^= ( const
dynamic_bitset<_WordT, _Alloc> &__rhs ) [inline]
```

Operations on dynamic_bitsets.

Parameters

<code>__rhs</code>	A same-sized dynamic_bitset.
--------------------	------------------------------

These should be self-explanatory.

Definition at line 794 of file dynamic_bitset.

```
5.996.3.28 template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset<_WordT, _Alloc>& std::tr2::dynamic_bitset<_WordT, _Alloc>::operator|= ( const
dynamic_bitset<_WordT, _Alloc> &__rhs ) [inline]
```

Operations on dynamic_bitsets.

Parameters

<code>__rhs</code>	A same-sized dynamic_bitset.
--------------------	------------------------------

These should be self-explanatory.

Definition at line 787 of file dynamic_bitset.

```
5.996.3.29 template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset<_WordT, _Alloc> std::tr2::dynamic_bitset<_WordT, _Alloc>::operator~ ( ) const
[inline]
```

See the no-argument flip().

Definition at line 919 of file dynamic_bitset.

```
5.996.3.30 template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> void
std::tr2::dynamic_bitset<_WordT, _Alloc>::push_back ( bool __bit ) [inline]
```

Push a bit onto the high end of the bitset.

Definition at line 728 of file dynamic_bitset.

```
5.996.3.31 template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset<_WordT, _Alloc>& std::tr2::dynamic_bitset<_WordT, _Alloc>::reset ( ) [inline]
```

Sets every bit to false.

Definition at line 872 of file dynamic_bitset.

```
5.996.3.32  template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
             dynamic_bitset<_WordT, _Alloc>& std::tr2::dynamic_bitset<_WordT, _Alloc>::reset ( size_type __pos )
             [inline]
```

Sets a given bit to false.

Parameters

<code>__pos</code>	The index of the bit.
--------------------	-----------------------

Exceptions

<code>std::out_of_range</code>	If <code>__pos</code> is bigger the size of the set.
--------------------------------	--

Same as writing `set(__pos, false)`.

Definition at line 886 of file `dynamic_bitset`.

5.996.3.33 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> void
std::tr2::dynamic_bitset<_WordT, _Alloc>::resize(size_type __nbits, bool __value = false) [inline]`

Resize the bitset.

Definition at line 705 of file `dynamic_bitset`.

Referenced by `std::tr2::operator>>()`.

5.996.3.34 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset<_WordT, _Alloc>& std::tr2::dynamic_bitset<_WordT, _Alloc>::set() [inline]`

Sets every bit to true.

Definition at line 847 of file `dynamic_bitset`.

5.996.3.35 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
dynamic_bitset<_WordT, _Alloc>& std::tr2::dynamic_bitset<_WordT, _Alloc>::set(size_type __pos, bool
__val=true) [inline]`

Sets a given bit to a particular value.

Parameters

<code>__pos</code>	The index of the bit.
<code>__val</code>	Either true or false, defaults to true.

Exceptions

<code>std::out_of_range</code>	If <code>__pos</code> is bigger the size of the set.
--------------------------------	--

Definition at line 861 of file `dynamic_bitset`.

5.996.3.36 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> size_type
std::tr2::dynamic_bitset<_WordT, _Alloc>::size () const [inline], [noexcept]`

Returns the total number of bits.

Definition at line 1007 of file `dynamic_bitset`.

Referenced by `std::tr2::operator>>()`.

5.996.3.37 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> void
std::tr2::dynamic_bitset<_WordT, _Alloc>::swap (dynamic_bitset<_WordT, _Alloc> &__b) [inline]`

Swap with another bitset.

Definition at line 665 of file `dynamic_bitset`.

5.996.3.38 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> bool
std::tr2::dynamic_bitset<_WordT, _Alloc>::test (size_type __pos) const [inline]`

Tests the value of a bit.

Parameters

<code>__pos</code>	The index of a bit.
--------------------	---------------------

Returns

The value at `__pos`.

Exceptions

<code>std::out_of_range</code>	If <code>__pos</code> is bigger the size of the set.
--------------------------------	--

Definition at line 1034 of file `dynamic_bitset`.

5.996.3.39 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> template<typename
_CharT = char, typename _Traits = std::char_traits<_CharT>, typename _Alloc1 = std::allocator<_CharT>>
std::basic_string<_CharT, _Traits, _Alloc1> std::tr2::dynamic_bitset<_WordT, _Alloc>::to_string (_CharT
__zero = _CharT('0'), _CharT __one = _CharT('1')) const [inline]`

Returns a character interpretation of the `dynamic_bitset`.

Returns

The string equivalent of the bits.

Note the ordering of the bits: decreasing character positions correspond to increasing bit positions (see the main class notes for an example).

Definition at line 972 of file `dynamic_bitset`.

5.996.3.40 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> unsigned long long
std::tr2::dynamic_bitset<_WordT, _Alloc >::to_ullong () const [inline]`

Returns a numerical interpretation of the `dynamic_bitset`.

Returns

The integral equivalent of the bits.

Exceptions

<code>std::overflow_error</code>	If there are too many bits to be represented in an <code>unsigned long</code> .
----------------------------------	---

Definition at line 957 of file `dynamic_bitset`.

5.996.3.41 `template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>> unsigned long
std::tr2::dynamic_bitset<_WordT, _Alloc >::to_ulong () const [inline]`

Returns a numerical interpretation of the `dynamic_bitset`.

Returns

The integral equivalent of the bits.

Exceptions

<code>std::overflow_error</code>	If there are too many bits to be represented in an <code>unsigned long</code> .
----------------------------------	---

Definition at line 947 of file `dynamic_bitset`.

The documentation for this class was generated from the following files:

- [dynamic_bitset](#)
- [dynamic_bitset.tcc](#)

5.997 `std::tr2::dynamic_bitset<_WordT, _Alloc >::reference` Class Reference

Public Member Functions

- **reference** ([dynamic_bitset](#) &__b, size_type __pos)
- [reference](#) & **flip** ()
- **operator bool** () const
- [reference](#) & **operator=** (bool __x)
- [reference](#) & **operator=** (const [reference](#) &__j)
- bool **operator~** () const

Friends

- class **dynamic_bitset**

5.997.1 Detailed Description

```
template<typename _WordT = unsigned long long, typename _Alloc = std::allocator<_WordT>>
class std::tr2::dynamic_bitset< _WordT, _Alloc >::reference
```

This encapsulates the concept of a single bit. An instance of this class is a proxy for an actual bit; this way the individual bit operations are done as faster word-size bitwise instructions.

Most users will never need to use this class directly; conversions to and from bool are automatic and should be transparent. Overloaded operators help to preserve the illusion.

(On a typical system, this "bit %reference" is 64 times the size of an actual bit. Ha.)

Definition at line 507 of file dynamic_bitset.

The documentation for this class was generated from the following file:

- [dynamic_bitset](#)

5.998 std::try_to_lock_t Struct Reference

5.998.1 Detailed Description

Try to acquire ownership of the mutex without blocking.

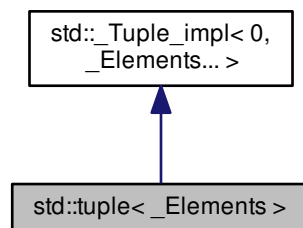
Definition at line 135 of file std_mutex.h.

The documentation for this struct was generated from the following file:

- [std_mutex.h](#)

5.999 std::tuple< _Elements > Class Template Reference

Inheritance diagram for std::tuple< _Elements >:



Public Types

- `template<typename _Dummy >`
`using _TCC = _TC< is_same< _Dummy, void >::value, _Elements... >`
- `template<typename... _UElements>`
`using _TMC = _TC<(sizeof...(_Elements)==sizeof...(_UElements)), _Elements... >`
- `template<typename _Dummy >`
`using _TNTC = _TC< is_same< _Dummy, void >::value &&sizeof...(_Elements)==1, _Elements... >`

Public Member Functions

- `template<typename _Dummy = void, typename enable_if< _TCC< _Dummy >::template _ConstructibleTuple< _Elements... >() &&_TCC< _Dummy >::template _ImplicitlyConvertibleTuple< _Elements... >() &&(sizeof...(_Elements) >= 1)>`
`constexpr tuple (const _Elements &...__elements)`
- `template<typename _Dummy = void, typename enable_if< _TCC< _Dummy >::template _ConstructibleTuple< _Elements... >() &&_TCC< _Dummy >::template _ImplicitlyConvertibleTuple< _Elements... >() &&(sizeof...(_Elements) >= 1)>`
`constexpr tuple (const _Elements &...__elements)`
- `template<typename... _UElements, typename enable_if< _TC< sizeof...(_UElements)==1, _Elements... >::template _NotSameTuple< _UElements... >() &&_TMC< _UElements... >::template _MoveConstructibleTuple< _UElements... >() &&_TMC< _UElements... >::template _ImplicitlyMoveConvertibleTuple< _UElements... >() &&(sizeof...(_Elements) >= 1)>`
`constexpr tuple (_UElements &&...__elements)`
- `template<typename... _UElements, typename enable_if< _TC< sizeof...(_UElements)==1, _Elements... >::template _NotSameTuple< _UElements... >() &&_TMC< _UElements... >::template _MoveConstructibleTuple< _UElements... >() &&!_TMC< _UElements... >::template _ImplicitlyMoveConvertibleTuple< _UElements... >() &&(sizeof...(_Elements) >= 1)>`
`constexpr tuple (_UElements &&...__elements)`
- `constexpr tuple (const tuple &)=default`
- `constexpr tuple (tuple &&)=default`
- `template<typename... _UElements, typename _Dummy = void, typename enable_if< _TMC< _UElements... >::template _ConstructibleTuple< _UElements... >() &&_TMC< _UElements... >::template _ImplicitlyConvertibleTuple< _UElements... >() &&_TNTC< _Dummy >::template _NonNestedTuple< const tuple< _UElements... > &>(), bool >::type = true>`
`constexpr tuple (const tuple< _UElements... > &__in)`
- `template<typename... _UElements, typename _Dummy = void, typename enable_if< _TMC< _UElements... >::template _ConstructibleTuple< _UElements... >() &&!_TMC< _UElements... >::template _ImplicitlyConvertibleTuple< _UElements... >() &&_TNTC< _Dummy >::template _NonNestedTuple< const tuple< _UElements... > &>(), bool >::type = false>`
`constexpr tuple (const tuple< _UElements... > &__in)`
- `template<typename... _UElements, typename _Dummy = void, typename enable_if< _TMC< _UElements... >::template _MoveConstructibleTuple< _UElements... >() &&_TMC< _UElements... >::template _ImplicitlyMoveConvertibleTuple< _UElements... >() &&_TNTC< _Dummy >::template _NonNestedTuple< tuple< _UElements... > &&>(), bool >::type = true>`
`constexpr tuple (tuple< _UElements... > &&__in)`
- `template<typename... _UElements, typename _Dummy = void, typename enable_if< _TMC< _UElements... >::template _MoveConstructibleTuple< _UElements... >() &&!_TMC< _UElements... >::template _ImplicitlyMoveConvertibleTuple< _UElements... >() &&_TNTC< _Dummy >::template _NonNestedTuple< tuple< _UElements... > &&>(), bool >::type = false>`
`constexpr tuple (tuple< _UElements... > &&__in)`
- `template<typename _Alloc >`
`tuple (allocator_arg_t __tag, const _Alloc &__a)`
- `template<typename _Alloc, typename _Dummy = void, typename enable_if< _TCC< _Dummy >::template _ConstructibleTuple< _Elements... >() &&_TCC< _Dummy >::template _ImplicitlyConvertibleTuple< _Elements... >(), bool >::type = true>`
`tuple (allocator_arg_t __tag, const _Alloc &__a, const _Elements &...__elements)`
- `template<typename _Alloc, typename _Dummy = void, typename enable_if< _TCC< _Dummy >::template _ConstructibleTuple< _Elements... >() &&!_TCC< _Dummy >::template _ImplicitlyConvertibleTuple< _Elements... >(), bool >::type = false>`
`tuple (allocator_arg_t __tag, const _Alloc &__a, const _Elements &...__elements)`

- `template<typename _Alloc , typename... _UElements, typename enable_if< _TMC< _UElements... >::template _MoveConstructible< Tuple< _UElements... >() &&_TMC< _UElements... >::template _ImplicitlyMoveConvertibleTuple< _UElements... >(), bool >::type = true>`
tuple ([allocator_arg_t](#) __tag, const _Alloc &__a, _UElements &&...__elements)
- `template<typename _Alloc , typename... _UElements, typename enable_if< _TMC< _UElements... >::template _MoveConstructible< Tuple< _UElements... >() &&!_TMC< _UElements... >::template _ImplicitlyMoveConvertibleTuple< _UElements... >(), bool >::type = false>`
tuple ([allocator_arg_t](#) __tag, const _Alloc &__a, _UElements &&...__elements)
- `template<typename _Alloc >`
tuple ([allocator_arg_t](#) __tag, const _Alloc &__a, const [tuple](#) &__in)
- `template<typename _Alloc >`
tuple ([allocator_arg_t](#) __tag, const _Alloc &__a, [tuple](#) &&__in)
- `template<typename _Alloc , typename... _UElements, typename enable_if< _TMC< _UElements... >::template _ConstructibleTuple< Tuple< _UElements... >() &&_TMC< _UElements... >::template _ImplicitlyConvertibleTuple< _UElements... >(), bool >::type = true>`
tuple ([allocator_arg_t](#) __tag, const _Alloc &__a, const [tuple](#)< _UElements... > &__in)
- `template<typename _Alloc , typename... _UElements, typename enable_if< _TMC< _UElements... >::template _ConstructibleTuple< Tuple< _UElements... >() &&!_TMC< _UElements... >::template _ImplicitlyConvertibleTuple< _UElements... >(), bool >::type = false>`
tuple ([allocator_arg_t](#) __tag, const _Alloc &__a, const [tuple](#)< _UElements... > &__in)
- `template<typename _Alloc , typename... _UElements, typename enable_if< _TMC< _UElements... >::template _MoveConstructible< Tuple< _UElements... >() &&_TMC< _UElements... >::template _ImplicitlyMoveConvertibleTuple< _UElements... >(), bool >::type = true>`
tuple ([allocator_arg_t](#) __tag, const _Alloc &__a, [tuple](#)< _UElements... > &&__in)
- `template<typename _Alloc , typename... _UElements, typename enable_if< _TMC< _UElements... >::template _MoveConstructible< Tuple< _UElements... >() &&!_TMC< _UElements... >::template _ImplicitlyMoveConvertibleTuple< _UElements... >(), bool >::type = false>`
tuple ([allocator_arg_t](#) __tag, const _Alloc &__a, [tuple](#)< _UElements... > &&__in)
- [tuple](#) & **operator=** (const [tuple](#) &__in)
- [tuple](#) & **operator=** ([tuple](#) &&__in) noexcept(is_nothrow_move_assignable< [Inherited](#) >::value)
- `template<typename... _UElements, typename = typename enable_if<sizeof...(_UElements) == sizeof...(_Elements)>::type>`
[tuple](#) & **operator=** (const [tuple](#)< _UElements... > &__in)
- `template<typename... _UElements, typename = typename enable_if<sizeof...(_UElements) == sizeof...(_Elements)>::type>`
[tuple](#) & **operator=** ([tuple](#)< _UElements... > &&__in)
- void **swap** ([tuple](#) &__in) noexcept(noexcept(__in._M_swap(__in)))

5.999.1 Detailed Description

```
template<typename... _Elements>
class std::tuple< _Elements >
```

Primary class template, [tuple](#).

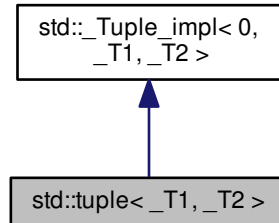
Definition at line 461 of file [tuple](#).

The documentation for this class was generated from the following file:

- [tuple](#)

5.1000 std::tuple< _T1, _T2 > Class Template Reference

Inheritance diagram for std::tuple< _T1, _T2 >:



Public Types

- template<typename _Dummy >
using **TCC** = _TC< is_same< _Dummy, void >::value, _T1, _T2 >
- using **TMC** = _TC< true, _T1, _T2 >

Public Member Functions

- template<typename _Dummy = void, typename enable_if< _TCC< _Dummy >::template _ConstructibleTuple< _T1, _T2 >() &&_TCC< _Dummy >::template _ImplicitlyConvertibleTuple< _T1, _T2 >(), bool >::type = true>
constexpr **tuple** (const _T1 &__a1, const _T2 &__a2)
- template<typename _Dummy = void, typename enable_if< _TCC< _Dummy >::template _ConstructibleTuple< _T1, _T2 >() &&!_TCC< _Dummy >::template _ImplicitlyConvertibleTuple< _T1, _T2 >(), bool >::type = false>
constexpr **tuple** (const _T1 &__a1, const _T2 &__a2)
- template<typename _U1, typename _U2, typename enable_if< _TMC::template _MoveConstructibleTuple< _U1, _U2 >() &&_TMC< _U1, _U2 >::template _ImplicitlyMoveConvertibleTuple< _U1, _U2 >(), bool >::type = true>
constexpr **tuple** (_U1 &&__a1, _U2 &&__a2)
- template<typename _U1, typename _U2, typename enable_if< _TMC::template _MoveConstructibleTuple< _U1, _U2 >() &&!_TMC< _U1, _U2 >::template _ImplicitlyMoveConvertibleTuple< _U1, _U2 >(), bool >::type = false>
constexpr **tuple** (_U1 &&__a1, _U2 &&__a2)
- constexpr **tuple** (const **tuple** &)=default
- constexpr **tuple** (**tuple** &&)=default
- template<typename _U1, typename _U2, typename enable_if< _TMC::template _ConstructibleTuple< _U1, _U2 >() &&_TMC::template _ImplicitlyConvertibleTuple< _U1, _U2 >(), bool >::type = true>
constexpr **tuple** (const **tuple**< _U1, _U2 > &__in)
- template<typename _U1, typename _U2, typename enable_if< _TMC::template _ConstructibleTuple< _U1, _U2 >() &&!_TMC::template _ImplicitlyConvertibleTuple< _U1, _U2 >(), bool >::type = false>
constexpr **tuple** (const **tuple**< _U1, _U2 > &__in)
- template<typename _U1, typename _U2, typename enable_if< _TMC::template _MoveConstructibleTuple< _U1, _U2 >() &&_TMC< _U1, _U2 >::template _ImplicitlyMoveConvertibleTuple< _U1, _U2 >(), bool >::type = true>
constexpr **tuple** (**tuple**< _U1, _U2 > &&__in)

- `template<typename _Alloc, typename _U1, typename _U2, typename enable_if<_TMC::template _MoveConstructibleTuple<_U1, _U2>() &&!_TMC::template _ImplicitlyMoveConvertibleTuple<_U1, _U2>(), bool >::type = false>`
`tuple (allocator_arg_t __tag, const _Alloc &__a, pair<_U1, _U2> &&__in)`
- `tuple & operator= (const tuple &__in)`
- `tuple & operator= (tuple &&__in) noexcept(is_nothrow_move_assignable<_Inherited>::value)`
- `template<typename _U1, typename _U2>`
`tuple & operator= (const tuple<_U1, _U2> &__in)`
- `template<typename _U1, typename _U2>`
`tuple & operator= (tuple<_U1, _U2> &&__in)`
- `template<typename _U1, typename _U2>`
`tuple & operator= (const pair<_U1, _U2> &__in)`
- `template<typename _U1, typename _U2>`
`tuple & operator= (pair<_U1, _U2> &&__in)`
- `void swap (tuple &__in) noexcept(noexcept(__in._M_swap(__in)))`

5.1000.1 Detailed Description

```
template<typename _T1, typename _T2>
class std::tuple<_T1, _T2>
```

Partial specialization, 2-element tuple. Includes construction and assignment from a pair.

Definition at line 866 of file tuple.

The documentation for this class was generated from the following file:

- [tuple](#)

5.1001 `std::tuple_element<_Int, _Tp>` Struct Template Reference

5.1001.1 Detailed Description

```
template<std::size_t _Int, typename _Tp>
struct std::tuple_element<_Int, _Tp>
```

`tuple_element`

Gives the type of the *ith* element of a given tuple type.

Definition at line 325 of file array.

The documentation for this struct was generated from the following file:

- [array](#)

5.1002 `std::tuple_element< 0, std::pair< _Tp1, _Tp2 > >` Struct Template Reference

Public Types

- `typedef _Tp1 type`

5.1002.1 Detailed Description

```
template<class _Tp1, class _Tp2>
struct std::tuple_element< 0, std::pair< _Tp1, _Tp2 > >
```

Partial specialization for `std::pair`.

Definition at line 151 of file `utility`.

The documentation for this struct was generated from the following file:

- [utility](#)

5.1003 `std::tuple_element< 0, tuple< _Head, _Tail... > >` Struct Template Reference

Public Types

- `typedef _Head type`

5.1003.1 Detailed Description

```
template<typename _Head, typename... _Tail>
struct std::tuple_element< 0, tuple< _Head, _Tail... > >
```

Basis case for `tuple_element`: The first element is the one we're seeking.

Definition at line 1231 of file `tuple`.

The documentation for this struct was generated from the following file:

- [tuple](#)

5.1004 `std::tuple_element< 1, std::pair< _Tp1, _Tp2 > >` Struct Template Reference

Public Types

- `typedef _Tp2 type`

5.1004.1 Detailed Description

```
template<class _Tp1, class _Tp2>
struct std::tuple_element< 1, std::pair< _Tp1, _Tp2 > >
```

Partial specialization for std::pair.

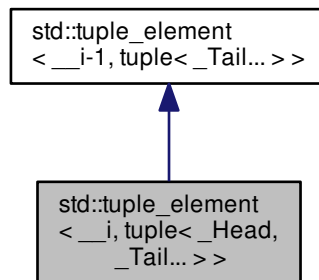
Definition at line 156 of file utility.

The documentation for this struct was generated from the following file:

- [utility](#)

5.1005 std::tuple_element< __i, tuple< _Head, _Tail... > > Struct Template Reference

Inheritance diagram for std::tuple_element< __i, tuple< _Head, _Tail... > >:



5.1005.1 Detailed Description

```
template<std::size_t __i, typename _Head, typename... _Tail>
struct std::tuple_element< __i, tuple< _Head, _Tail... > >
```

Recursive case for tuple_element: strip off the first element in the tuple and retrieve the (i-1)th element of the remaining tuple.

Definition at line 1224 of file tuple.

The documentation for this struct was generated from the following file:

- [tuple](#)

5.1006 `std::tuple_element<_Int, std::__debug::array<_Tp, _Nm>>` Struct Template Reference

Public Types

- `typedef _Tp type`

5.1006.1 Detailed Description

```
template<std::size_t _Int, typename _Tp, std::size_t _Nm>
struct std::tuple_element<_Int, std::__debug::array<_Tp, _Nm>>
```

`tuple_element`

Definition at line 307 of file `debug/array`.

The documentation for this struct was generated from the following file:

- [debug/array](#)

5.1007 `std::tuple_element<_Int,::array<_Tp, _Nm>>` Struct Template Reference

Public Types

- `typedef _Tp type`

5.1007.1 Detailed Description

```
template<std::size_t _Int, typename _Tp, std::size_t _Nm>
struct std::tuple_element<_Int,::array<_Tp, _Nm>>
```

Partial specialization for `std::array`.

Definition at line 329 of file `array`.

The documentation for this struct was generated from the following file:

- [array](#)

5.1008 std::tuple_size< _Tp > Struct Template Reference

5.1008.1 Detailed Description

```
template<typename _Tp>
struct std::tuple_size< _Tp >
```

tuple_size

Finds the size of a given tuple type.

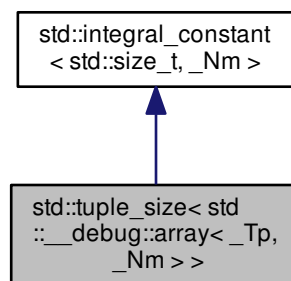
Definition at line 316 of file array.

The documentation for this struct was generated from the following file:

- [array](#)

5.1009 std::tuple_size< std::__debug::array< _Tp, _Nm > > Struct Template Reference

Inheritance diagram for std::tuple_size< std::__debug::array< _Tp, _Nm > >:



Public Types

- typedef [integral_constant](#)< std::size_t, __v > **type**
- typedef std::size_t **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr std::size_t **value**

5.1009.1 Detailed Description

```
template<typename _Tp, std::size_t _Nm>
struct std::tuple_size< std::__debug::array< _Tp, _Nm > >
```

tuple_size

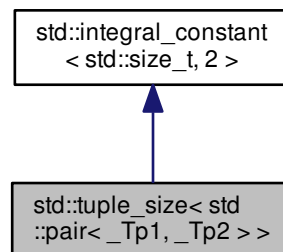
Definition at line 302 of file debug/array.

The documentation for this struct was generated from the following file:

- [debug/array](#)

5.1010 std::tuple_size< std::pair< _Tp1, _Tp2 > > Struct Template Reference

Inheritance diagram for std::tuple_size< std::pair< _Tp1, _Tp2 > >:



Public Types

- typedef [integral_constant](#)< std::size_t, __v > **type**
- typedef std::size_t **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr `std::size_t` **value**

5.1010.1 Detailed Description

```
template<class _Tp1, class _Tp2>
struct std::tuple_size< std::pair< _Tp1, _Tp2 > >
```

Partial specialization for `std::pair`.

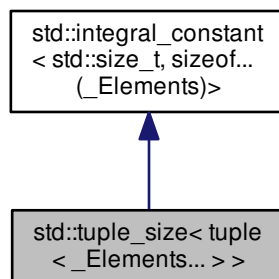
Definition at line 146 of file `utility`.

The documentation for this struct was generated from the following file:

- [utility](#)

5.1011 `std::tuple_size< tuple< _Elements... > >` Struct Template Reference

Inheritance diagram for `std::tuple_size< tuple< _Elements... > >`:



Public Types

- typedef [integral_constant](#)< `std::size_t`, `__v` > **type**
- typedef `std::size_t` **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr std::size_t **value**

5.1011.1 Detailed Description

```
template<typename... _Elements>
struct std::tuple_size< tuple< _Elements... > >
```

class tuple_size

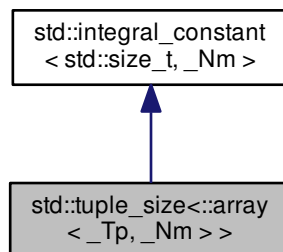
Definition at line 1238 of file tuple.

The documentation for this struct was generated from the following file:

- [tuple](#)

5.1012 std::tuple_size<::array< _Tp, _Nm > > Struct Template Reference

Inheritance diagram for std::tuple_size<::array< _Tp, _Nm > >:



Public Types

- typedef [integral_constant](#)< std::size_t, __v > **type**
- typedef std::size_t **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr std::size_t **value**

5.1012.1 Detailed Description

```
template<typename _Tp, std::size_t _Nm>
struct std::tuple_size<::array< _Tp, _Nm > >
```

Partial specialization for std::array.

Definition at line 320 of file array.

The documentation for this struct was generated from the following file:

- [array](#)

5.1013 std::type_index Struct Reference

Public Member Functions

- **type_index** (const [type_info](#) &__rhs) noexcept
- size_t **hash_code** () const noexcept
- const char * **name** () const noexcept
- bool **operator!=** (const [type_index](#) &__rhs) const noexcept
- bool **operator<** (const [type_index](#) &__rhs) const noexcept
- bool **operator<=** (const [type_index](#) &__rhs) const noexcept
- bool **operator==** (const [type_index](#) &__rhs) const noexcept
- bool **operator>** (const [type_index](#) &__rhs) const noexcept
- bool **operator>=** (const [type_index](#) &__rhs) const noexcept

5.1013.1 Detailed Description

Class type_index

The class type_index provides a simple wrapper for type_info which can be used as an index type in associative containers (23.6) and in unordered associative containers (23.7).

Definition at line 52 of file typeindex.

The documentation for this struct was generated from the following file:

- [typeindex](#)

5.1014 std::type_info Class Reference

Inherited by `__cxxabiv1::__array_type_info`, `__cxxabiv1::__class_type_info`, `__cxxabiv1::__enum_type_info`, `__cxxabiv1::__function_type_info`, `__cxxabiv1::__fundamental_type_info`, and `__cxxabiv1::__pbase_type_info`.

Public Member Functions

- virtual `~type_info()`
- virtual `bool __do_catch (const type_info * __thr_type, void ** __thr_obj, unsigned __outer) const`
- virtual `bool __do_upcast (const __cxxabiv1::__class_type_info * __target, void ** __obj_ptr) const`
- virtual `bool __is_function_p () const`
- virtual `bool __is_pointer_p () const`
- `bool before (const type_info & __arg) const noexcept`
- `size_t hash_code () const noexcept`
- `const char * name () const noexcept`
- `bool operator!= (const type_info & __arg) const noexcept`
- `bool operator== (const type_info & __arg) const noexcept`

Protected Member Functions

- `type_info (const char * __n)`

Protected Attributes

- `const char * __name`

5.1014.1 Detailed Description

Part of RTTI.

The `type_info` class describes type information generated by an implementation.

Definition at line 88 of file `typeinfo`.

5.1014.2 Constructor & Destructor Documentation

5.1014.2.1 virtual `std::type_info::~~type_info ()` [virtual]

Destructor first. Being the first non-inline virtual function, this controls in which translation unit the vtable is emitted. The compiler makes use of that information to know where to emit the runtime-mandated `type_info` structures in the new-abi.

5.1014.3 Member Function Documentation

5.1014.3.1 const char* std::type_info::name () const [inline],[noexcept]

Returns an *implementation-defined* byte string; this is not portable between compilers!

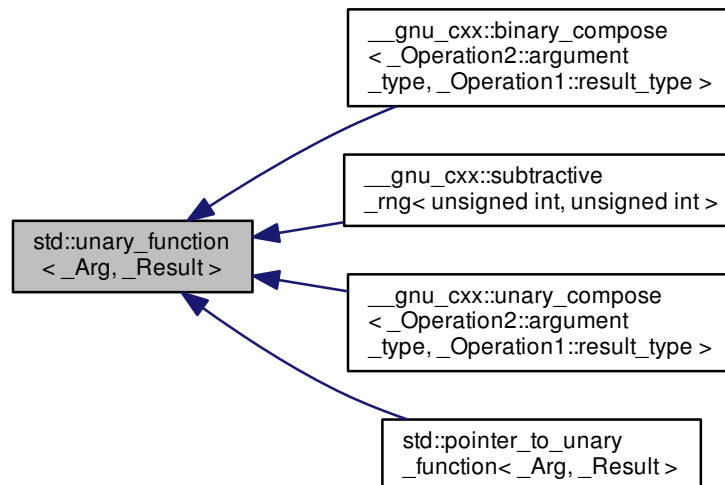
Definition at line 99 of file typeinfo.

The documentation for this class was generated from the following file:

- [typeinfo](#)

5.1015 std::unary_function< _Arg, _Result > Struct Template Reference

Inheritance diagram for std::unary_function< _Arg, _Result >:



Public Types

- typedef _Arg [argument_type](#)
- typedef _Result [result_type](#)

5.1015.1 Detailed Description

```
template<typename _Arg, typename _Result>
struct std::unary_function< _Arg, _Result >
```

This is one of the [functor base classes](#).

Definition at line 105 of file stl_function.h.

5.1015.2 Member Typedef Documentation

5.1015.2.1 `template<typename _Arg, typename _Result> typedef _Arg std::unary_function< _Arg, _Result >::argument_type`

`argument_type` is the type of the argument

Definition at line 108 of file `stl_function.h`.

5.1015.2.2 `template<typename _Arg, typename _Result> typedef _Result std::unary_function< _Arg, _Result >::result_type`

`result_type` is the return type

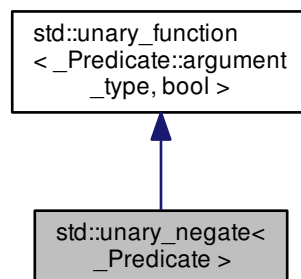
Definition at line 111 of file `stl_function.h`.

The documentation for this struct was generated from the following file:

- [stl_function.h](#)

5.1016 `std::unary_negate< _Predicate >` Class Template Reference

Inheritance diagram for `std::unary_negate< _Predicate >`:



Public Types

- typedef `_Predicate::argument_type` [argument_type](#)
- typedef `bool` [result_type](#)

Public Member Functions

- `_GLIBCXX14_CONSTEXPR unary_negate` (const _Predicate &__x)
- `_GLIBCXX14_CONSTEXPR bool operator()` (const typename _Predicate::argument_type &__x) const

Protected Attributes

- `_Predicate _M_pred`

5.1016.1 Detailed Description

```
template<typename _Predicate>
class std::unary_negate<_Predicate>
```

One of the [negation functors](#).

Definition at line 741 of file `stl_function.h`.

5.1016.2 Member Typedef Documentation

5.1016.2.1 `typedef _Predicate::argument_type std::unary_function<_Predicate::argument_type, bool>::argument_type` `[inherited]`

`argument_type` is the type of the argument

Definition at line 108 of file `stl_function.h`.

5.1016.2.2 `typedef bool std::unary_function<_Predicate::argument_type, bool>::result_type` `[inherited]`

`result_type` is the return type

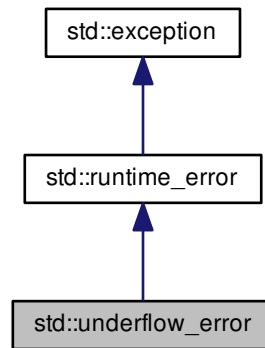
Definition at line 111 of file `stl_function.h`.

The documentation for this class was generated from the following file:

- [stl_function.h](#)

5.1017 std::underflow_error Class Reference

Inheritance diagram for std::underflow_error:



Public Member Functions

- **underflow_error** (const [string](#) &__arg) _GLIBCXX_TXN_SAFE
- **underflow_error** (const char *) _GLIBCXX_TXN_SAFE
- virtual const char * [what](#) () const _GLIBCXX_TXN_SAFE_DYN noexcept

5.1017.1 Detailed Description

Thrown to indicate arithmetic underflow.

Definition at line 252 of file stdexcept.

5.1017.2 Member Function Documentation

5.1017.2.1 virtual const char* std::runtime_error::what () const [virtual],[noexcept],[inherited]

Returns a C-style character string describing the general cause of the current error (the same string passed to the ctor).

Reimplemented from [std::exception](#).

The documentation for this class was generated from the following file:

- [stdexcept](#)

5.1018 std::uniform_int_distribution< _IntType > Class Template Reference

Classes

- struct [param_type](#)

Public Types

- typedef _IntType [result_type](#)

Public Member Functions

- [uniform_int_distribution](#) (_IntType __a=0, _IntType __b=std::numeric_limits< _IntType >::max())
- [uniform_int_distribution](#) (const [param_type](#) &__p)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void [__generate](#) (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void [__generate](#) (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- template<typename _UniformRandomNumberGenerator >
void [__generate](#) ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [result_type](#) [a](#) () const
- [result_type](#) [b](#) () const
- [result_type](#) [max](#) () const
- [result_type](#) [min](#) () const
- template<typename _UniformRandomNumberGenerator >
[result_type](#) [operator\(\)](#) (_UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
[result_type](#) [operator\(\)](#) (_UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [param_type](#) [param](#) () const
- void [param](#) (const [param_type](#) &__param)
- void [reset](#) ()

Friends

- bool [operator==](#) (const [uniform_int_distribution](#) &__d1, const [uniform_int_distribution](#) &__d2)

5.1018.1 Detailed Description

```
template<typename _IntType = int>
class std::uniform_int_distribution< _IntType >
```

Uniform discrete distribution for random numbers. A discrete random distribution on the range $[min, max]$ with equal probability throughout the range.

Definition at line 61 of file uniform_int_dist.h.

5.1018.2 Member Typedef Documentation

5.1018.2.1 `template<typename _IntType = int> typedef _IntType std::uniform_int_distribution< _IntType >::result_type`

The type of the range of the distribution.

Definition at line 64 of file `uniform_int_dist.h`.

5.1018.3 Constructor & Destructor Documentation

5.1018.3.1 `template<typename _IntType = int> std::uniform_int_distribution< _IntType >::uniform_int_distribution (_IntType __a = 0, _IntType __b = std::numeric_limits<_IntType>::max()) [inline], [explicit]`

Constructs a uniform distribution object.

Definition at line 104 of file `uniform_int_dist.h`.

5.1018.4 Member Function Documentation

5.1018.4.1 `template<typename _IntType = int> result_type std::uniform_int_distribution< _IntType >::max () const [inline]`

Returns the inclusive upper bound of the distribution range.

Definition at line 156 of file `uniform_int_dist.h`.

5.1018.4.2 `template<typename _IntType = int> result_type std::uniform_int_distribution< _IntType >::min () const [inline]`

Returns the inclusive lower bound of the distribution range.

Definition at line 149 of file `uniform_int_dist.h`.

5.1018.4.3 `template<typename _IntType = int> template<typename _UniformRandomNumberGenerator > result_type std::uniform_int_distribution< _IntType >::operator() (_UniformRandomNumberGenerator & __urng) [inline]`

Generating functions.

Definition at line 164 of file `uniform_int_dist.h`.

5.1018.4.4 `template<typename _IntType = int> param_type std::uniform_int_distribution< _IntType >::param () const [inline]`

Returns the parameter set of the distribution.

Definition at line 134 of file `uniform_int_dist.h`.

Referenced by `std::operator>>()`.

5.1018.4.5 `template<typename _IntType = int> void std::uniform_int_distribution< _IntType >::param (const param_type & __param) [inline]`

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

Definition at line 142 of file `uniform_int_dist.h`.

5.1018.4.6 `template<typename _IntType = int> void std::uniform_int_distribution<_IntType>::reset () [inline]`

Resets the distribution state.

Does nothing for the uniform integer distribution.

Definition at line 120 of file `uniform_int_dist.h`.

5.1018.5 Friends And Related Function Documentation

5.1018.5.1 `template<typename _IntType = int> bool operator==(const uniform_int_distribution<_IntType> &__d1, const uniform_int_distribution<_IntType> &__d2) [friend]`

Return true if two uniform integer distributions have the same parameters.

Definition at line 199 of file `uniform_int_dist.h`.

The documentation for this class was generated from the following file:

- [uniform_int_dist.h](#)

5.1019 `std::uniform_int_distribution<_IntType>::param_type` Struct Reference

Public Types

- typedef [uniform_int_distribution<_IntType>](#) **distribution_type**

Public Member Functions

- **param_type** (`_IntType __a=0, _IntType __b=std::numeric_limits<_IntType>::max()`)
- **result_type a** () const
- **result_type b** () const

Friends

- bool **operator==** (const [param_type](#) &__p1, const [param_type](#) &__p2)

5.1019.1 Detailed Description

```
template<typename _IntType = int>
struct std::uniform_int_distribution< _IntType >::param_type
```

Parameter type.

Definition at line 70 of file uniform_int_dist.h.

The documentation for this struct was generated from the following file:

- [uniform_int_dist.h](#)

5.1020 std::uniform_real_distribution< _RealType > Class Template Reference

Classes

- struct [param_type](#)

Public Types

- typedef _RealType [result_type](#)

Public Member Functions

- [uniform_real_distribution](#) (_RealType __a=_RealType(0), _RealType __b=_RealType(1))
- **uniform_real_distribution** (const [param_type](#) &__p)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng)
- template<typename _ForwardIterator, typename _UniformRandomNumberGenerator >
void **generate** (_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- template<typename _UniformRandomNumberGenerator >
void **generate** ([result_type](#) *__f, [result_type](#) *__t, _UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [result_type](#) **a** () const
- [result_type](#) **b** () const
- [result_type](#) **max** () const
- [result_type](#) **min** () const
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng)
- template<typename _UniformRandomNumberGenerator >
[result_type](#) **operator()** (_UniformRandomNumberGenerator &__urng, const [param_type](#) &__p)
- [param_type](#) **param** () const
- void **param** (const [param_type](#) &__param)
- void **reset** ()

Friends

- bool `operator==` (const `uniform_real_distribution` &__d1, const `uniform_real_distribution` &__d2)

5.1020.1 Detailed Description

```
template<typename _RealType = double>
class std::uniform_real_distribution< _RealType >
```

Uniform continuous distribution for random numbers.

A continuous random distribution on the range [min, max) with equal probability throughout the range. The URNG should be real-valued and deliver number in the range [0, 1).

Definition at line 1702 of file random.h.

5.1020.2 Member Typedef Documentation

5.1020.2.1 `template<typename _RealType = double> typedef _RealType std::uniform_real_distribution< _RealType >::result_type`

The type of the range of the distribution.

Definition at line 1705 of file random.h.

5.1020.3 Constructor & Destructor Documentation

5.1020.3.1 `template<typename _RealType = double> std::uniform_real_distribution< _RealType >::uniform_real_distribution (_RealType __a = _RealType(0), _RealType __b = _RealType(1))`
`[inline], [explicit]`

Constructs a `uniform_real_distribution` object.

Parameters

<code>↔</code> __a	[IN] The lower bound of the distribution.
<code>↔</code> __b	[IN] The upper bound of the distribution.

Definition at line 1748 of file random.h.

5.1020.4 Member Function Documentation

5.1020.4.1 `template<typename _RealType = double> result_type std::uniform_real_distribution< _RealType >::max ()`
`const [inline]`

Returns the inclusive upper bound of the distribution range.

Definition at line 1800 of file random.h.

5.1020.4.2 `template<typename _RealType = double> result_type std::uniform_real_distribution< _RealType >::min ()`
`const [inline]`

Returns the inclusive lower bound of the distribution range.

Definition at line 1793 of file random.h.

5.1020.4.3 `template<typename _RealType = double> template<typename _UniformRandomNumberGenerator > result_type`
`std::uniform_real_distribution< _RealType >::operator() (_UniformRandomNumberGenerator & __urng)`
`[inline]`

Generating functions.

Definition at line 1808 of file random.h.

5.1020.4.4 `template<typename _RealType = double> param_type std::uniform_real_distribution< _RealType >::param (`
`) const [inline]`

Returns the parameter set of the distribution.

Definition at line 1778 of file random.h.

Referenced by `std::operator>>()`.

5.1020.4.5 `template<typename _RealType = double> void std::uniform_real_distribution< _RealType >::param (const`
`param_type & __param) [inline]`

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

Definition at line 1786 of file random.h.

5.1020.4.6 `template<typename _RealType = double> void std::uniform_real_distribution< _RealType >::reset ()`
`[inline]`

Resets the distribution state.

Does nothing for the uniform real distribution.

Definition at line 1764 of file random.h.

5.1020.5 Friends And Related Function Documentation

5.1020.5.1 `template<typename _RealType = double> bool operator==(const uniform_real_distribution<_RealType> &__d1, const uniform_real_distribution<_RealType> &__d2)` [*friend*]

Return true if two uniform real distributions have the same parameters.

Definition at line 1848 of file `random.h`.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.1021 `std::uniform_real_distribution<_RealType>::param_type` Struct Reference

Public Types

- typedef `uniform_real_distribution<_RealType>` **distribution_type**

Public Member Functions

- **param_type** (`_RealType __a=_RealType(0), _RealType __b=_RealType(1)`)
- **result_type a** () const
- **result_type b** () const

Friends

- bool **operator==** (const `param_type` &__p1, const `param_type` &__p2)

5.1021.1 Detailed Description

```
template<typename _RealType = double>
struct std::uniform_real_distribution<_RealType>::param_type
```

Parameter type.

Definition at line 1711 of file `random.h`.

The documentation for this struct was generated from the following file:

- [random.h](#)

5.1022 `std::unique_lock<_Mutex>` Class Template Reference

Public Types

- typedef `_Mutex` **`mutex_type`**

Public Member Functions

- **`unique_lock`** (`mutex_type` &__m)
- **`unique_lock`** (`mutex_type` &__m, [defer_lock_t](#)) noexcept
- **`unique_lock`** (`mutex_type` &__m, [try_to_lock_t](#))
- **`unique_lock`** (`mutex_type` &__m, [adopt_lock_t](#))
- template<typename `_Clock` , typename `_Duration` >
`unique_lock` (`mutex_type` &__m, const [chrono::time_point](#)< `_Clock`, `_Duration` > &__atime)
- template<typename `_Rep` , typename `_Period` >
`unique_lock` (`mutex_type` &__m, const [chrono::duration](#)< `_Rep`, `_Period` > &__rtime)
- **`unique_lock`** (const [unique_lock](#) &)=delete
- **`unique_lock`** ([unique_lock](#) &&__u) noexcept
- void **`lock`** ()
- `mutex_type` * **`mutex`** () const noexcept
- **`operator bool`** () const noexcept
- [unique_lock](#) & **`operator=`** (const [unique_lock](#) &)=delete
- [unique_lock](#) & **`operator=`** ([unique_lock](#) &&__u) noexcept
- bool **`owns_lock`** () const noexcept
- `mutex_type` * **`release`** () noexcept
- void **`swap`** ([unique_lock](#) &__u) noexcept
- bool **`try_lock`** ()
- template<typename `_Rep` , typename `_Period` >
bool **`try_lock_for`** (const [chrono::duration](#)< `_Rep`, `_Period` > &__rtime)
- template<typename `_Clock` , typename `_Duration` >
bool **`try_lock_until`** (const [chrono::time_point](#)< `_Clock`, `_Duration` > &__atime)
- void **`unlock`** ()

5.1022.1 Detailed Description

```
template<typename _Mutex>
class std::unique_lock<_Mutex>
```

A movable scoped lock type.

A `unique_lock` controls mutex ownership within a scope. Ownership of the mutex can be delayed until after construction and can be transferred to another `unique_lock` by move construction or move assignment. If a mutex lock is owned when the destructor runs ownership will be released.

Definition at line 185 of file `std_mutex.h`.

The documentation for this class was generated from the following file:

- [std_mutex.h](#)

5.1023 std::unique_ptr< _Tp, _Dp > Class Template Reference

Public Types

- template<typename _Up, typename _Ep >
using **__safe_conversion_up** = __and_< is_convertible< typename [unique_ptr](#)< _Up, _Ep >::pointer, pointer >, __not_< [is_array](#)< _Up >>, __or_< __and_< [is_reference](#)< deleter_type >, is_same< deleter_type, _Ep >>, __and_< __not_< [is_reference](#)< deleter_type >>, is_convertible< _Ep, deleter_type >> > >
- typedef _Dp **deleter_type**
- typedef _Tp **element_type**
- typedef _Pointer::type **pointer**

Public Member Functions

- constexpr [unique_ptr](#) () noexcept
- [unique_ptr](#) (pointer __p) noexcept
- [unique_ptr](#) (pointer __p, typename conditional< [is_reference](#)< deleter_type >::value, deleter_type, const deleter_type & >::type __d) noexcept
- [unique_ptr](#) (pointer __p, typename remove_reference< deleter_type >::type &&__d) noexcept
- constexpr [unique_ptr](#) (nullptr_t) noexcept
- [unique_ptr](#) ([unique_ptr](#) &&__u) noexcept
- template<typename _Up, typename _Ep, typename = _Require< __safe_conversion_up< _Up, _Ep >, typename conditional<is_←reference<_Dp>::value, is_same<_Ep, _Dp>, is_convertible<_Ep, _Dp>>::type>>
[unique_ptr](#) ([unique_ptr](#)< _Up, _Ep > &&__u) noexcept
- template<typename _Up, typename >
unique_ptr ([auto_ptr](#)< _Up > &&__u) noexcept
- **unique_ptr** (const [unique_ptr](#) &)=delete
- [~unique_ptr](#) () noexcept
- pointer [get](#) () const noexcept
- deleter_type & [get_deleter](#) () noexcept
- const deleter_type & [get_deleter](#) () const noexcept
- [operator bool](#) () const noexcept
- add_lvalue_reference< element_type >::type [operator*](#) () const
- pointer [operator->](#) () const noexcept
- [unique_ptr](#) & [operator=](#) ([unique_ptr](#) &&__u) noexcept
- template<typename _Up, typename _Ep >
enable_if< __and_< __safe_conversion_up< _Up, _Ep >, is_assignable< deleter_type &, _Ep && > >::value,
[unique_ptr](#) & >::type [operator=](#) ([unique_ptr](#)< _Up, _Ep > &&__u) noexcept
- [unique_ptr](#) & [operator=](#) (nullptr_t) noexcept
- [unique_ptr](#) & [operator=](#) (const [unique_ptr](#) &)=delete
- pointer [release](#) () noexcept
- void [reset](#) (pointer __p=pointer()) noexcept
- void [swap](#) ([unique_ptr](#) &__u) noexcept

5.1023.1 Detailed Description

```
template<typename _Tp, typename _Dp = default_delete<_Tp>>
class std::unique_ptr< _Tp, _Dp >
```

20.7.1.2 unique_ptr for single objects.

Definition at line 116 of file unique_ptr.h.

5.1023.2 Constructor & Destructor Documentation

5.1023.2.1 `template<typename _Tp, typename _Dp = default_delete<_Tp>> constexpr std::unique_ptr<_Tp, _Dp>::unique_ptr () [inline], [noexcept]`

Default constructor, creates a unique_ptr that owns nothing.

Definition at line 158 of file unique_ptr.h.

Referenced by `std::auto_ptr<_Tp>::auto_ptr()`.

5.1023.2.2 `template<typename _Tp, typename _Dp = default_delete<_Tp>> std::unique_ptr<_Tp, _Dp>::unique_ptr (pointer __p) [inline], [explicit], [noexcept]`

Takes ownership of a pointer.

Parameters

<code>__p</code>	A pointer to an object of <code>element_type</code>
------------------	---

The deleter will be value-initialized.

Definition at line 170 of file unique_ptr.h.

5.1023.2.3 `template<typename _Tp, typename _Dp = default_delete<_Tp>> std::unique_ptr<_Tp, _Dp>::unique_ptr (pointer __p, typename conditional<is_reference< deleter_type >::value, deleter_type, const deleter_type &>::type __d) [inline], [noexcept]`

Takes ownership of a pointer.

Parameters

<code>__p</code>	A pointer to an object of <code>element_type</code>
<code>__d</code>	A reference to a deleter.

The deleter will be initialized with `__d`

Definition at line 185 of file unique_ptr.h.

5.1023.2.4 `template<typename _Tp, typename _Dp = default_delete<_Tp>> std::unique_ptr<_Tp, _Dp>::unique_ptr (pointer __p, typename remove_reference< deleter_type >::type && __d) [inline], [noexcept]`

Takes ownership of a pointer.

Parameters

\leftrightarrow _p	A pointer to an object of <code>element_type</code>
\leftrightarrow _d	An rvalue reference to a deleter.

The deleter will be initialized with `std::move(__d)`

Definition at line 197 of file `unique_ptr.h`.

5.1023.2.5 `template<typename _Tp, typename _Dp = default_delete<_Tp>> constexpr std::unique_ptr< _Tp, _Dp >::unique_ptr(nullptr_t) [inline], [noexcept]`

Creates a `unique_ptr` that owns nothing.

Definition at line 204 of file `unique_ptr.h`.

5.1023.2.6 `template<typename _Tp, typename _Dp = default_delete<_Tp>> std::unique_ptr< _Tp, _Dp >::unique_ptr(unique_ptr< _Tp, _Dp > && __u) [inline], [noexcept]`

Move constructor.

Definition at line 209 of file `unique_ptr.h`.

5.1023.2.7 `template<typename _Tp, typename _Dp = default_delete<_Tp>> template<typename _Up, typename _Ep, typename = _Require< __safe_conversion_up<_Up, _Ep>, typename conditional<is_reference<_Dp>::value, is_same<_Ep, _Dp>, is_convertible<_Ep, _Dp>>::type>> std::unique_ptr< _Tp, _Dp >::unique_ptr(unique_ptr< _Up, _Ep > && __u) [inline], [noexcept]`

Converting constructor from another type.

Requires that the pointer owned by `__u` is convertible to the type of pointer owned by this object, `__u` does not own an array, and `__u` has a compatible deleter type.

Definition at line 223 of file `unique_ptr.h`.

5.1023.2.8 `template<typename _Tp, typename _Dp = default_delete<_Tp>> std::unique_ptr< _Tp, _Dp >::~unique_ptr() [inline], [noexcept]`

Destructor, invokes the deleter if the stored pointer is not null.

Definition at line 235 of file `unique_ptr.h`.

5.1023.3 Member Function Documentation

5.1023.3.1 `template<typename _Tp, typename _Dp = default_delete<_Tp>> pointer std::unique_ptr<_Tp, _Dp>::get (void) const [inline], [noexcept]`

Return the stored pointer.

Definition at line 307 of file `unique_ptr.h`.

Referenced by `std::unique_ptr<_Tp[], _Dp>::swap()`.

5.1023.3.2 `template<typename _Tp, typename _Dp = default_delete<_Tp>> deleter_type& std::unique_ptr<_Tp, _Dp>::get_deleter () [inline], [noexcept]`

Return a reference to the stored deleter.

Definition at line 312 of file `unique_ptr.h`.

5.1023.3.3 `template<typename _Tp, typename _Dp = default_delete<_Tp>> const deleter_type& std::unique_ptr<_Tp, _Dp>::get_deleter () const [inline], [noexcept]`

Return a reference to the stored deleter.

Definition at line 317 of file `unique_ptr.h`.

5.1023.3.4 `template<typename _Tp, typename _Dp = default_delete<_Tp>> std::unique_ptr<_Tp, _Dp>::operator bool () const [inline], [explicit], [noexcept]`

Return `true` if the stored pointer is not null.

Definition at line 321 of file `unique_ptr.h`.

5.1023.3.5 `template<typename _Tp, typename _Dp = default_delete<_Tp>> add_lvalue_reference<element_type>::type std::unique_ptr<_Tp, _Dp>::operator* () const [inline]`

Dereference the stored pointer.

Definition at line 291 of file `unique_ptr.h`.

5.1023.3.6 `template<typename _Tp, typename _Dp = default_delete<_Tp>> pointer std::unique_ptr<_Tp, _Dp>::operator-> () const [inline], [noexcept]`

Return the stored pointer.

Definition at line 299 of file `unique_ptr.h`.

5.1023.3.7 `template<typename _Tp, typename _Dp = default_delete<_Tp>> unique_ptr& std::unique_ptr<_Tp, _Dp>::operator= (unique_ptr<_Tp, _Dp> && __u) [inline], [noexcept]`

Move assignment operator.

Parameters

<code>_↔</code>	The object to transfer ownership from.
<code>_u</code>	

Invokes the deleter first if this object owns a pointer.

Definition at line 252 of file unique_ptr.h.

```
5.1023.3.8  template<typename _Tp, typename _Dp = default_delete<_Tp>> template<typename _Up, typename _Ep
> enable_if< __and< __safe_conversion_up<_Up, _Ep>, is_assignable<deleter_type&, _Ep&&> >::value,
unique_ptr&>::type std::unique_ptr<_Tp, _Dp>::operator= ( unique_ptr<_Up, _Ep> && _u )
[inline], [noexcept]
```

Assignment from another type.

Parameters

<code>_↔</code>	The object to transfer ownership from, which owns a convertible pointer to a non-array object.
<code>_u</code>	

Invokes the deleter first if this object owns a pointer.

Definition at line 272 of file unique_ptr.h.

```
5.1023.3.9  template<typename _Tp, typename _Dp = default_delete<_Tp>> unique_ptr& std::unique_ptr<_Tp, _Dp
>::operator= ( nullptr_t ) [inline], [noexcept]
```

Reset the unique_ptr to empty, invoking the deleter if necessary.

Definition at line 281 of file unique_ptr.h.

```
5.1023.3.10 template<typename _Tp, typename _Dp = default_delete<_Tp>> pointer std::unique_ptr<_Tp, _Dp>::release (
) [inline], [noexcept]
```

Release ownership of any stored pointer.

Definition at line 328 of file unique_ptr.h.

```
5.1023.3.11 template<typename _Tp, typename _Dp = default_delete<_Tp>> void std::unique_ptr<_Tp, _Dp>::reset (
pointer __p = pointer() ) [inline], [noexcept]
```

Replace the stored pointer.

Parameters

<code>_↔</code>	The new pointer to store.
<code>_p</code>	

The deleter will be invoked if a pointer is already owned.

Definition at line 342 of file `unique_ptr.h`.

5.1023.3.12 `template<typename _Tp, typename _Dp = default_delete<_Tp>> void std::unique_ptr<_Tp, _Dp>::swap (unique_ptr<_Tp, _Dp> &__u) [inline], [noexcept]`

Exchange the pointer and deleter with another object.

Definition at line 352 of file `unique_ptr.h`.

Referenced by `std::unique_ptr<_Tp[], _Dp>::swap()`.

The documentation for this class was generated from the following files:

- [unique_ptr.h](#)
- [auto_ptr.h](#)

5.1024 `std::unique_ptr<_Tp[], _Dp>` Class Template Reference

Public Types

- `template<typename _Up>`
using `__safe_conversion_raw` = `__and< __or< __or< is_same< _Up, pointer >, is_same< _Up, nullptr_t >>, __and< is_pointer< _Up >, is_same< pointer, element_type * >, is_convertible< typename remove_`
`pointer< _Up >::type(*)[], element_type(*)[]> > >`
- `template<typename _Up, typename _Ep, typename _Up_up = unique_ptr<_Up, _Ep>, typename _Up_element_type = typename _Up`
`__up::element_type>`
using `__safe_conversion_up` = `__and< is_array< _Up >, is_same< pointer, element_type * >, is_`
`same< typename _Up_up::pointer, _Up_element_type * >, is_convertible< _Up_element_type(*)[], element_`
`_type(*)[]>, __or< __and< is_reference< deleter_type >, is_same< deleter_type, _Ep >>, __and< __`
`not< is_reference< deleter_type >>, is_convertible< _Ep, deleter_type >>> >`
- typedef `_Dp deleter_type`
- typedef `_Tp element_type`
- typedef `_Pointer::type pointer`

Public Member Functions

- constexpr `unique_ptr()` `noexcept`
- `template<typename _Up, typename = typename enable_if< __safe_conversion_raw<_Up>::value, bool>::type>`
`unique_ptr(_Up __p) noexcept`
- `template<typename _Up, typename = typename enable_if< __safe_conversion_raw<_Up>::value, bool>::type>`
`unique_ptr(_Up __p, typename conditional< is_reference< deleter_type >::value, deleter_type, const deleter_`
`_type &>::type __d) noexcept`
- `template<typename _Up, typename = typename enable_if< __safe_conversion_raw<_Up>::value, bool>::type>`
`unique_ptr(_Up __p, typename remove_reference< deleter_type >::type &&__d) noexcept`
- `unique_ptr(unique_ptr &&__u) noexcept`
- constexpr `unique_ptr(nullptr_t) noexcept`

- `template<typename _Up, typename _Ep, typename = _Require<__safe_conversion_up<_Up, _Ep>>>`
`unique_ptr (unique_ptr< _Up, _Ep > &&__u) noexcept`
- `unique_ptr (const unique_ptr &)=delete`
- `~unique_ptr ()`
- `pointer get () const noexcept`
- `deleter_type & get_deleter () noexcept`
- `const deleter_type & get_deleter () const noexcept`
- `operator bool () const noexcept`
- `unique_ptr & operator= (unique_ptr &&__u) noexcept`
- `template<typename _Up, typename _Ep >`
`enable_if< __and< __safe_conversion_up< _Up, _Ep >, is_assignable< deleter_type &, _Ep && > >::value,`
`unique_ptr & >::type operator= (unique_ptr< _Up, _Ep > &&__u) noexcept`
- `unique_ptr & operator= (nullptr_t) noexcept`
- `unique_ptr & operator= (const unique_ptr &)=delete`
- `std::add_lvalue_reference< element_type >::type operator[] (size_t __i) const`
- `pointer release () noexcept`
- `template<typename _Up, typename = _Require< __or_<is_same<_Up, pointer>, __and<is_same<pointer, element_type*>, is_`
`pointer<_Up>, is_convertible< typename remove_pointer<_Up>::type(*)[], element_type(*)[] > > > >>`
`void reset (_Up __p) noexcept`
- `void reset (nullptr_t=nullptr) noexcept`
- `void swap (unique_ptr &__u) noexcept`

5.1024.1 Detailed Description

```
template<typename _Tp, typename _Dp>
class std::unique_ptr< _Tp[], _Dp >
```

20.7.1.3 unique_ptr for array objects with a runtime length

Definition at line 368 of file unique_ptr.h.

5.1024.2 Constructor & Destructor Documentation

5.1024.2.1 `template<typename _Tp, typename _Dp > constexpr std::unique_ptr< _Tp[], _Dp >::unique_ptr ()`
`[inline], [noexcept]`

Default constructor, creates a unique_ptr that owns nothing.

Definition at line 435 of file unique_ptr.h.

5.1024.2.2 `template<typename _Tp, typename _Dp > template<typename _Up, typename = typename enable_if<`
`__safe_conversion_raw<_Up>::value, bool>::type> std::unique_ptr< _Tp[], _Dp >::unique_ptr (_Up __p)`
`[inline], [explicit], [noexcept]`

Takes ownership of a pointer.

Parameters

\leftrightarrow _p	A pointer to an array of a type safely convertible to an array of <code>element_type</code>
-------------------------	---

The deleter will be value-initialized.

Definition at line 451 of file `unique_ptr.h`.

```
5.1024.2.3 template<typename _Tp , typename _Dp > template<typename _Up , typename = typename enable_if<
    __safe_conversion_raw<_Up>::value, bool>::type> std::unique_ptr< _Tp[], _Dp >::unique_ptr ( _Up __p,
    typename conditional< is_reference< deleter_type >::value, deleter_type, const deleter_type & >::type __d )
    [inline], [noexcept]
```

Takes ownership of a pointer.

Parameters

\leftrightarrow _p	A pointer to an array of a type safely convertible to an array of <code>element_type</code>
\leftrightarrow _d	A reference to a deleter.

The deleter will be initialized with `__d`

Definition at line 467 of file `unique_ptr.h`.

```
5.1024.2.4 template<typename _Tp , typename _Dp > template<typename _Up , typename = typename enable_if<
    __safe_conversion_raw<_Up>::value, bool>::type> std::unique_ptr< _Tp[], _Dp >::unique_ptr ( _Up __p,
    typename remove_reference< deleter_type >::type && __d ) [inline], [noexcept]
```

Takes ownership of a pointer.

Parameters

\leftrightarrow _p	A pointer to an array of a type safely convertible to an array of <code>element_type</code>
\leftrightarrow _d	A reference to a deleter.

The deleter will be initialized with `std::move(__d)`

Definition at line 483 of file `unique_ptr.h`.

```
5.1024.2.5 template<typename _Tp , typename _Dp > std::unique_ptr< _Tp[], _Dp >::unique_ptr ( unique_ptr< _Tp[],
    _Dp > && __u ) [inline], [noexcept]
```

Move constructor.

Definition at line 490 of file `unique_ptr.h`.

5.1024.2.6 `template<typename _Tp, typename _Dp> constexpr std::unique_ptr<_Tp[],_Dp>::unique_ptr(nullptr_t)`
`[inline], [noexcept]`

Creates a `unique_ptr` that owns nothing.

Definition at line 494 of file `unique_ptr.h`.

5.1024.2.7 `template<typename _Tp, typename _Dp> std::unique_ptr<_Tp[],_Dp>::~~unique_ptr()` `[inline]`

Destructor, invokes the deleter if the stored pointer is not null.

Definition at line 503 of file `unique_ptr.h`.

References `std::get_deleter()`.

5.1024.3 Member Function Documentation

5.1024.3.1 `template<typename _Tp, typename _Dp> pointer std::unique_ptr<_Tp[],_Dp>::get(void) const`
`[inline], [noexcept]`

Return the stored pointer.

Definition at line 567 of file `unique_ptr.h`.

5.1024.3.2 `template<typename _Tp, typename _Dp> deleter_type& std::unique_ptr<_Tp[],_Dp>::get_deleter()`
`[inline], [noexcept]`

Return a reference to the stored deleter.

Definition at line 572 of file `unique_ptr.h`.

5.1024.3.3 `template<typename _Tp, typename _Dp> const deleter_type& std::unique_ptr<_Tp[],_Dp>::get_deleter()`
`const [inline], [noexcept]`

Return a reference to the stored deleter.

Definition at line 577 of file `unique_ptr.h`.

5.1024.3.4 `template<typename _Tp, typename _Dp> std::unique_ptr<_Tp[],_Dp>::operator bool() const` `[inline],`
`[explicit], [noexcept]`

Return `true` if the stored pointer is not null.

Definition at line 581 of file `unique_ptr.h`.

5.1024.3.5 `template<typename _Tp, typename _Dp> unique_ptr& std::unique_ptr<_Tp[],_Dp>::operator=(`
`unique_ptr<_Tp[],_Dp> && __u)` `[inline], [noexcept]`

Move assignment operator.

Parameters

<code>_↔</code>	The object to transfer ownership from.
<code>_u</code>	

Invokes the deleter first if this object owns a pointer.

Definition at line 520 of file `unique_ptr.h`.

References `std::get_deleter()`.

```
5.1024.3.6  template<typename _Tp , typename _Dp > template<typename _Up , typename _Ep > enable_if<__and<_↔
             _safe_conversion_up<_Up, _Ep>, is_assignable<deleter_type&, _Ep&&> >::value, unique_ptr&>::type
             std::unique_ptr<_Tp[], _Dp>::operator= ( unique_ptr<_Up, _Ep> && _u )  [inline], [noexcept]
```

Assignment from another type.

Parameters

<code>_↔</code>	The object to transfer ownership from, which owns a convertible pointer to an array object.
<code>_u</code>	

Invokes the deleter first if this object owns a pointer.

Definition at line 540 of file `unique_ptr.h`.

References `std::get_deleter()`.

```
5.1024.3.7  template<typename _Tp , typename _Dp > unique_ptr& std::unique_ptr<_Tp[], _Dp>::operator= ( nullptr_t )
             [inline], [noexcept]
```

Reset the `unique_ptr` to empty, invoking the deleter if necessary.

Definition at line 549 of file `unique_ptr.h`.

```
5.1024.3.8  template<typename _Tp , typename _Dp > std::add_lvalue_reference<element_type>::type std::unique_ptr<
             _Tp[], _Dp>::operator[] ( size_t __i ) const  [inline]
```

Access an element of owned array.

Definition at line 559 of file `unique_ptr.h`.

```
5.1024.3.9  template<typename _Tp , typename _Dp > pointer std::unique_ptr<_Tp[], _Dp>::release ( )  [inline],
             [noexcept]
```



Release ownership of any stored pointer.

Definition at line 588 of file `unique_ptr.h`.

```
5.1024.3.10 template<typename _Tp , typename _Dp > template<typename _Up , typename = _Require<__or<is_same<_Up,
             pointer>, __and<is_same<pointer, element_type*>, is_pointer<_Up>, is_convertible< typename
             remove_pointer<_Up>::type(*)[], element_type(*)[] > > > > void std::unique_ptr<_Tp[], _Dp>::reset ( _Up
             __p )  [inline], [noexcept]
```

Replace the stored pointer.

Parameters

	The new pointer to store.
	

The deleter will be invoked if a pointer is already owned.

Definition at line 614 of file unique_ptr.h.

References `std::get_deleter()`.

5.1024.3.11 `template<typename _Tp, typename _Dp> void std::unique_ptr<_Tp[], _Dp>::swap (unique_ptr<_Tp[], _Dp> &__u) [inline], [noexcept]`

Exchange the pointer and deleter with another object.

Definition at line 630 of file unique_ptr.h.

References `std::unique_ptr<_Tp, _Dp>::get()`, and `std::unique_ptr<_Tp, _Dp>::swap()`.

The documentation for this class was generated from the following file:

- [unique_ptr.h](#)

5.1025 std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc > Class Template Reference

Public Types

- `typedef _Hashtable::key_type` [key_type](#)
- `typedef _Hashtable::value_type` [value_type](#)
- `typedef _Hashtable::mapped_type` [mapped_type](#)
- `typedef _Hashtable::hasher` [hasher](#)
- `typedef _Hashtable::key_equal` [key_equal](#)
- `typedef _Hashtable::allocator_type` [allocator_type](#)
- `typedef _Hashtable::pointer` [pointer](#)
- `typedef _Hashtable::const_pointer` [const_pointer](#)
- `typedef _Hashtable::reference` [reference](#)
- `typedef _Hashtable::const_reference` [const_reference](#)
- `typedef _Hashtable::iterator` [iterator](#)
- `typedef _Hashtable::const_iterator` [const_iterator](#)
- `typedef _Hashtable::local_iterator` [local_iterator](#)
- `typedef _Hashtable::const_local_iterator` [const_local_iterator](#)
- `typedef _Hashtable::size_type` [size_type](#)
- `typedef _Hashtable::difference_type` [difference_type](#)

Public Member Functions

- `unordered_map` ()=default
- `unordered_map` (`size_type` __n, const `hasher` &__hf=`hasher`(), const `key_equal` &__eq=`key_equal`(), const `allocator_type` &__a=`allocator_type`())
- `template<typename _InputIterator >`
`unordered_map` (_InputIterator __first, _InputIterator __last, `size_type` __n=0, const `hasher` &__hf=`hasher`(), const `key_equal` &__eq=`key_equal`(), const `allocator_type` &__a=`allocator_type`())
- `unordered_map` (const `unordered_map` &)=default
- `unordered_map` (`unordered_map` &&)=default
- `unordered_map` (const `allocator_type` &__a)
- `unordered_map` (const `unordered_map` &__umap, const `allocator_type` &__a)
- `unordered_map` (`unordered_map` &&__umap, const `allocator_type` &__a)
- `unordered_map` (`initializer_list`< `value_type` > __l, `size_type` __n=0, const `hasher` &__hf=`hasher`(), const `key_←`
`equal` &__eq=`key_equal`(), const `allocator_type` &__a=`allocator_type`())
- `unordered_map` (`size_type` __n, const `allocator_type` &__a)
- `unordered_map` (`size_type` __n, const `hasher` &__hf, const `allocator_type` &__a)
- `template<typename _InputIterator >`
`unordered_map` (_InputIterator __first, _InputIterator __last, `size_type` __n, const `allocator_type` &__a)
- `template<typename _InputIterator >`
`unordered_map` (_InputIterator __first, _InputIterator __last, `size_type` __n, const `hasher` &__hf, const `allocator_type` &__a)
- `unordered_map` (`initializer_list`< `value_type` > __l, `size_type` __n, const `allocator_type` &__a)
- `unordered_map` (`initializer_list`< `value_type` > __l, `size_type` __n, const `hasher` &__hf, const `allocator_type` &←
__a)
- `iterator begin` () noexcept
- `local_iterator begin` (`size_type` __n)
- `size_type bucket` (const `key_type` &__key) const
- `size_type bucket_count` () const noexcept
- `size_type bucket_size` (`size_type` __n) const
- `void clear` () noexcept
- `size_type count` (const `key_type` &__x) const
- `template<typename... _Args>`
`std::pair`< `iterator`, bool > `emplace` (_Args &&...__args)
- `template<typename... _Args>`
`iterator emplace_hint` (const `iterator` __pos, _Args &&...__args)
- `bool empty` () const noexcept
- `iterator end` () noexcept
- `local_iterator end` (`size_type` __n)
- `size_type erase` (const `key_type` &__x)
- `iterator erase` (const `iterator` __first, const `iterator` __last)
- `allocator_type get_allocator` () const noexcept
- `hasher hash_function` () const
- `template<typename _InputIterator >`
`void insert` (_InputIterator __first, _InputIterator __last)
- `void insert` (`initializer_list`< `value_type` > __l)
- `key_equal key_eq` () const
- `float load_factor` () const noexcept
- `size_type max_bucket_count` () const noexcept
- `float max_load_factor` () const noexcept
- `void max_load_factor` (float __z)

- `size_type max_size ()` const noexcept
- `unordered_map & operator= (const unordered_map &)=default`
- `unordered_map & operator= (unordered_map &&)=default`
- `unordered_map & operator= (initializer_list< value_type > __l)`
- `void rehash (size_type __n)`
- `void reserve (size_type __n)`
- `size_type size ()` const noexcept
- `void swap (unordered_map &__x)` noexcept(noexcept(__M_h.swap(__x._M_h)))
- `const_iterator begin ()` const noexcept
- `const_iterator cbegin ()` const noexcept
- `const_iterator end ()` const noexcept
- `const_iterator cend ()` const noexcept
- `std::pair< iterator, bool > insert (const value_type &__x)`
- `template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type> std::pair< iterator, bool > insert (_Pair &&__x)`
- `iterator insert (const_iterator __hint, const value_type &__x)`
- `template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type> iterator insert (const_iterator __hint, _Pair &&__x)`
- `iterator erase (const_iterator __position)`
- `iterator erase (iterator __position)`
- `iterator find (const key_type &__x)`
- `const_iterator find (const key_type &__x)` const
- `std::pair< iterator, iterator > equal_range (const key_type &__x)`
- `std::pair< const_iterator, const_iterator > equal_range (const key_type &__x)` const
- `mapped_type & operator[] (const key_type &__k)`
- `mapped_type & operator[] (key_type &&__k)`
- `mapped_type & at (const key_type &__k)`
- `const mapped_type & at (const key_type &__k)` const
- `const_local_iterator begin (size_type __n)` const
- `const_local_iterator cbegin (size_type __n)` const
- `const_local_iterator end (size_type __n)` const
- `const_local_iterator cend (size_type __n)` const

Friends

- `template<typename _Key1, typename _Tp1, typename _Hash1, typename _Pred1, typename _Alloc1 >
bool operator== (const unordered_map< _Key1, _Tp1, _Hash1, _Pred1, _Alloc1 > &, const unordered_map<
_Key1, _Tp1, _Hash1, _Pred1, _Alloc1 > &)`

5.1025.1 Detailed Description

```
template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::
::allocator<std::pair<const _Key, _Tp> >>
class std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >
```

A standard container composed of unique keys (containing at most one of each key value) that associates values of another type with the keys.

Template Parameters

<code>_Key</code>	Type of key objects.
<code>_Tp</code>	Type of mapped objects.
<code>_Hash</code>	Hashing function object type, defaults to <code>hash<_Value></code> .
<code>_Pred</code>	Predicate function object type, defaults to <code>equal_to<_Value></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>std::allocator<std::pair<const _Key, _Tp>></code> .

Meets the requirements of a [container](#), and [unordered associative container](#)

The resulting value type of the container is `std::pair<const _Key, _Tp>`.

Base is `_Hashtable`, dispatched at compile time via template alias `__umap_hashtable`.

Definition at line 98 of file `unordered_map.h`.

5.1025.2 Member Typedef Documentation

5.1025.2.1 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp> >> typedef _Hashtable::allocator_type std::unordered_map< _Key,
_Tp, _Hash, _Pred, _Alloc >::allocator_type`

Public typedefs.

Definition at line 112 of file `unordered_map.h`.

5.1025.2.2 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp> >> typedef _Hashtable::const_iterator std::unordered_map< _Key,
_Tp, _Hash, _Pred, _Alloc >::const_iterator`

Iterator-related typedefs.

Definition at line 122 of file `unordered_map.h`.

5.1025.2.3 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::const_local_iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::const_local_iterator`

Iterator-related typedefs.

Definition at line 124 of file `unordered_map.h`.

5.1025.2.4 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::const_pointer std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::const_pointer`

Iterator-related typedefs.

Definition at line 118 of file `unordered_map.h`.

5.1025.2.5 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::const_reference std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::const_reference`

Iterator-related typedefs.

Definition at line 120 of file `unordered_map.h`.

5.1025.2.6 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::difference_type std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::difference_type`

Iterator-related typedefs.

Definition at line 126 of file `unordered_map.h`.

5.1025.2.7 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::hasher std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::hasher`

Public typedefs.

Definition at line 110 of file `unordered_map.h`.

5.1025.2.8 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::iterator`

Iterator-related typedefs.

Definition at line 121 of file `unordered_map.h`.

5.1025.2.9 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::key_equal std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::key_equal`

Public typedefs.

Definition at line 111 of file `unordered_map.h`.

5.1025.2.10 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::key_type std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::key_type`

Public typedefs.

Definition at line 107 of file `unordered_map.h`.

5.1025.2.11 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::local_iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::local_iterator`

Iterator-related typedefs.

Definition at line 123 of file `unordered_map.h`.

5.1025.2.12 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::mapped_type std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::mapped_type`

Public typedefs.

Definition at line 109 of file `unordered_map.h`.

5.1025.2.13 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::pointer std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::pointer`

Iterator-related typedefs.

Definition at line 117 of file `unordered_map.h`.

5.1025.2.14 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::reference std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::reference`

Iterator-related typedefs.

Definition at line 119 of file `unordered_map.h`.

5.1025.2.15 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::size_type std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::size_type`

Iterator-related typedefs.

Definition at line 125 of file unordered_map.h.

5.1025.2.16 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::value_type std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::value_type`

Public typedefs.

Definition at line 108 of file unordered_map.h.

5.1025.3 Constructor & Destructor Documentation

5.1025.3.1 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::unordered_map() [default]`

Default constructor.

Referenced by `std::unordered_map<_Key, _Tp, _Hash, _Pred>::unordered_map()`.

5.1025.3.2 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::unordered_map(size_type __n, const hasher & __hf = hasher(), const key_equal & __eqf = key_equal(), const allocator_type & __a = allocator_type()) [inline],[explicit]`

Default constructor creates no elements.

Parameters

<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eqf</code>	A key equality functor.
<code>__a</code>	An allocator object.

Definition at line 142 of file unordered_map.h.

5.1025.3.3 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> template<typename _InputIterator> std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::unordered_map(_InputIterator __first, _InputIterator __last, size_type __n = 0, const hasher & __hf = hasher(), const key_equal & __eqf = key_equal(), const allocator_type & __a = allocator_type()) [inline]`

Builds an unordered_map from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eqf</code>	A key equality functor.
<code>__a</code>	An allocator object.

Create an `unordered_map` consisting of copies of the elements from `[__first,__last)`. This is linear in `N` (where `N` is `distance(__first,__last)`).

Definition at line 163 of file `unordered_map.h`.

```
5.1025.3.4  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc
            = std::allocator<std::pair<const _Key, _Tp>>> std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc
            >::unordered_map( const unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc> & ) [default]
```

Copy constructor.

```
5.1025.3.5  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc
            = std::allocator<std::pair<const _Key, _Tp>>> std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc
            >::unordered_map( unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc> && ) [default]
```

Move constructor.

```
5.1025.3.6  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc
            = std::allocator<std::pair<const _Key, _Tp>>> std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc
            >::unordered_map( const allocator_type & __a ) [inline],[explicit]
```

Creates an `unordered_map` with no elements.

Parameters

<code>__a</code>	An allocator object.
------------------	----------------------

Definition at line 182 of file `unordered_map.h`.

```
5.1025.3.7  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc
            = std::allocator<std::pair<const _Key, _Tp>>> std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc
            >::unordered_map( initializer_list<value_type> __l, size_type __n = 0, const hasher & __hf =
            hasher(), const key_equal & __eqf = key_equal(), const allocator_type & __a = allocator_type() )
            [inline]
```

Builds an `unordered_map` from an `initializer_list`.

Parameters

<code>__l</code>	An initializer_list.
<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eqf</code>	A key equality functor.
<code>__a</code>	An allocator object.

Create an unordered_map consisting of copies of the elements in the list. This is linear in N (where N is `__l.size()`).

Definition at line 217 of file unordered_map.h.

5.1025.4 Member Function Documentation

5.1025.4.1 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> mapped_type& std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::at(const key_type &__k) [inline]`

Access to unordered_map data.

Parameters

<code>__k</code>	The key for which data should be retrieved.
------------------	---

Returns

A reference to the data whose key is equal to `__k`, if such a data is present in the unordered_map.

Exceptions

<code>std::out_of_range</code>	If no such data is present.
--------------------------------	-----------------------------

Definition at line 920 of file unordered_map.h.

5.1025.4.2 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> const mapped_type& std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::at(const key_type &__k) const [inline]`

Access to unordered_map data.

Parameters

<code>__k</code>	The key for which data should be retrieved.
------------------	---

Returns

A reference to the data whose key is equal to `__k`, if such a data is present in the `unordered_map`.

Exceptions

<code>std::out_of_range</code>	If no such data is present.
--------------------------------	-----------------------------

Definition at line 924 of file `unordered_map.h`.

```
5.1025.4.3  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
             std::allocator<std::pair<const _Key, _Tp>>> iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc
             >::begin ( ) [inline], [noexcept]
```

Returns a read/write iterator that points to the first element in the `unordered_map`.

Definition at line 316 of file `unordered_map.h`.

```
5.1025.4.4  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
             std::allocator<std::pair<const _Key, _Tp>>> const_iterator std::unordered_map<_Key, _Tp, _Hash, _Pred,
             _Alloc >::begin ( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the `unordered_map`.

Definition at line 325 of file `unordered_map.h`.

```
5.1025.4.5  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
             std::allocator<std::pair<const _Key, _Tp>>> local_iterator std::unordered_map<_Key, _Tp, _Hash, _Pred,
             _Alloc >::begin ( size_type __n ) [inline]
```

Returns a read/write iterator pointing to the first bucket element.

Parameters

<code>__n</code>	The bucket index.
------------------	-------------------

Returns

A read/write local iterator.

Definition at line 965 of file `unordered_map.h`.

```
5.1025.4.6  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
             std::allocator<std::pair<const _Key, _Tp>>> const_local_iterator std::unordered_map<_Key, _Tp, _Hash,
             _Pred, _Alloc >::begin ( size_type __n ) const [inline]
```

Returns a read-only (constant) iterator pointing to the first bucket element.

Parameters

\leftrightarrow	The bucket index.
n	

Returns

A read-only local iterator.

Definition at line 976 of file unordered_map.h.

```
5.1025.4.7 template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> size_type std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc
>::bucket_count( ) const [inline], [noexcept]
```

Returns the number of buckets of the unordered_map.

Definition at line 932 of file unordered_map.h.

```
5.1025.4.8 template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> const_iterator std::unordered_map<_Key, _Tp, _Hash, _Pred,
_Alloc>::cbegin( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the unordered_map.

Definition at line 329 of file unordered_map.h.

```
5.1025.4.9 template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> const_local_iterator std::unordered_map<_Key, _Tp, _Hash,
_Pred, _Alloc>::cbegin( size_type n ) const [inline]
```

Returns a read-only (constant) iterator pointing to the first bucket element.

Parameters

\leftrightarrow	The bucket index.
n	

Returns

A read-only local iterator.

Definition at line 980 of file unordered_map.h.

```
5.1025.4.10 template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> const_iterator std::unordered_map<_Key, _Tp, _Hash, _Pred,
_Alloc>::cend( ) const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the unordered_map.

Definition at line 351 of file unordered_map.h.

```
5.1025.4.11  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
               std::allocator<std::pair<const _Key, _Tp>>> const_local_iterator std::unordered_map< _Key, _Tp, _Hash,
               _Pred, _Alloc >::cend ( size_type __n ) const    [inline]
```

Returns a read-only (constant) iterator pointing to one past the last bucket elements.

Parameters

$_n$	The bucket index.
-------	-------------------

Returns

A read-only local iterator.

Definition at line 1006 of file unordered_map.h.

```
5.1025.4.12  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
               std::allocator<std::pair<const _Key, _Tp>>> void std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc
               >::clear ( )    [inline], [noexcept]
```

Erases all elements in an unordered_map. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 803 of file unordered_map.h.

```
5.1025.4.13  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
               std::allocator<std::pair<const _Key, _Tp>>> size_type std::unordered_map< _Key, _Tp, _Hash, _Pred,
               _Alloc >::count ( const key_type &__x ) const    [inline]
```

Finds the number of elements.

Parameters

$_x$	Key to count.
-------	---------------

Returns

Number of elements with specified key.

This function only makes sense for unordered_multimap; for unordered_map the result will either be 0 (not present) or 1 (present).

Definition at line 868 of file unordered_map.h.

```
5.1025.4.14 template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> template<typename... _Args> std::pair<iterator, bool>
std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::emplace ( _Args &&... __args ) [inline]
```

Attempts to build and insert a std::pair into the unordered_map.

Parameters

<code>__args</code>	Arguments used to generate a new pair instance (see std::piecewise_construct for passing arguments to each part of the pair constructor).
---------------------	---

Returns

A pair, of which the first element is an iterator that points to the possibly inserted pair, and the second is a bool that is true if the pair was actually inserted.

This function attempts to build and insert a (key, value) pair into the unordered_map. An unordered_map relies on unique keys and thus a pair is only inserted if its first element (the key) is not already present in the unordered_map.

Insertion requires amortized constant time.

Definition at line 379 of file unordered_map.h.

Referenced by std::unordered_map<_Key, _Tp, _Hash, _Pred>::emplace_hint(), and std::unordered_map<_Key, _Tp, _Hash, _Pred>::insert().

```
5.1025.4.15 template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> template<typename... _Args> iterator std::unordered_map<
_Key, _Tp, _Hash, _Pred, _Alloc>::emplace_hint ( const_iterator __pos, _Args &&... __args ) [inline]
```

Attempts to build and insert a std::pair into the unordered_map.

Parameters

<code>__pos</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__args</code>	Arguments used to generate a new pair instance (see std::piecewise_construct for passing arguments to each part of the pair constructor).

Returns

An iterator that points to the element with key of the std::pair built from __args (may or may not be that std::pair).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument emplace() does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints for more on *hinting*.

Insertion requires amortized constant time.

Definition at line 410 of file unordered_map.h.

Referenced by `std::unordered_map<_Key, _Tp, _Hash, _Pred >::emplace_hint()`, and `std::unordered_map<_Key, _Tp, _Hash, _Pred >::insert()`.

```
5.1025.4.16  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
               std::allocator<std::pair<const _Key, _Tp> >> bool std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc
               >::empty( ) const    [inline], [noexcept]
```

Returns true if the unordered_map is empty.

Definition at line 296 of file unordered_map.h.

```
5.1025.4.17  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
               std::allocator<std::pair<const _Key, _Tp> >> iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc
               >::end( )    [inline], [noexcept]
```

Returns a read/write iterator that points one past the last element in the unordered_map.

Definition at line 338 of file unordered_map.h.

Referenced by `std::unordered_map<_Key, _Tp, _Hash, _Pred >::emplace_hint()`, and `std::unordered_map<_Key, _Tp, _Hash, _Pred >::insert()`.

```
5.1025.4.18  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
               std::allocator<std::pair<const _Key, _Tp> >> const_iterator std::unordered_map<_Key, _Tp, _Hash, _Pred,
               _Alloc >::end( ) const    [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the unordered_map.

Definition at line 347 of file unordered_map.h.

```
5.1025.4.19  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
               std::allocator<std::pair<const _Key, _Tp> >> local_iterator std::unordered_map<_Key, _Tp, _Hash, _Pred,
               _Alloc >::end( size_type __n )    [inline]
```

Returns a read/write iterator pointing to one past the last bucket elements.

Parameters

<code>__n</code>	The bucket index.
------------------	-------------------

Returns

A read/write local iterator.

Definition at line 991 of file unordered_map.h.

5.1025.4.20 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> const_local_iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::end (size_type __n) const [inline]`

Returns a read-only (constant) iterator pointing to one past the last bucket elements.

Parameters

<code>__n</code>	The bucket index.
------------------	-------------------

Returns

A read-only local iterator.

Definition at line 1002 of file unordered_map.h.

5.1025.4.21 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> std::pair<iterator, iterator> std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::equal_range (const key_type & __x) [inline]`

Finds a subsequence matching given key.

Parameters

<code>__x</code>	Key to be located.
------------------	--------------------

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function probably only makes sense for unordered_multimap.

Definition at line 881 of file unordered_map.h.

5.1025.4.22 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> std::pair<const_iterator, const_iterator> std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc >::equal_range (const key_type & __x) const [inline]`

Finds a subsequence matching given key.

Parameters

<code>__x</code>	Key to be located.
------------------	--------------------

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function probably only makes sense for `unordered_multimap`.

Definition at line 885 of file `unordered_map.h`.

```
5.1025.4.23  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
               std::allocator<std::pair<const _Key, _Tp>>> iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc
               >::erase ( const_iterator __position ) [inline]
```

Erases an element from an `unordered_map`.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following `__position` prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from an `unordered_map`. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 753 of file `unordered_map.h`.

```
5.1025.4.24  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
               std::allocator<std::pair<const _Key, _Tp>>> iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc
               >::erase ( iterator __position ) [inline]
```

Erases an element from an `unordered_map`.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following `__position` prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from an `unordered_map`. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 758 of file `unordered_map.h`.

5.1025.4.25 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> size_type std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::erase(const key_type & __x) [inline]`

Erases elements according to the provided key.

Parameters

<code>__x</code>	Key of element to be erased.
------------------	------------------------------

Returns

The number of elements erased.

This function erases all the elements located by the given key from an `unordered_map`. For an `unordered_map` the result of this function can only be 0 (not present) or 1 (present). Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 775 of file `unordered_map.h`.

5.1025.4.26 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::erase(const_iterator __first, const_iterator __last) [inline]`

Erases a [`__first`,`__last`) range of elements from an `unordered_map`.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be erased.
<code>__last</code>	Iterator pointing to the end of the range to be erased.

Returns

The iterator `__last`.

This function erases a sequence of elements from an `unordered_map`. Note that this function only erases the elements, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 793 of file `unordered_map.h`.

5.1025.4.27 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::find(const key_type & __x) [inline]`

Tries to locate an element in an `unordered_map`.

Parameters

$_ \leftrightarrow$	Key to be located.
$_x$	

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (`end()`) iterator.

Definition at line 850 of file `unordered_map.h`.

Referenced by `std::unordered_map<_Key, _Tp, _Hash, _Pred>::emplace_hint()`, and `std::unordered_map<_Key, _Tp, _Hash, _Pred>::insert()`.

5.1025.4.28 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> const_iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::find (const key_type & __x) const [inline]`

Tries to locate an element in an `unordered_map`.

Parameters

$_ \leftrightarrow$	Key to be located.
$_x$	

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (`end()`) iterator.

Definition at line 854 of file `unordered_map.h`.

5.1025.4.29 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> allocator_type std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::get_allocator () const [inline], [noexcept]`

Returns the allocator object with which the `unordered_map` was constructed.

Definition at line 289 of file `unordered_map.h`.

5.1025.4.30 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> hasher std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::hash_function () const [inline]`

Returns the hash functor object with which the `unordered_map` was constructed.

Definition at line 826 of file `unordered_map.h`.

5.1025.4.31 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> std::pair<iterator, bool> std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::insert (const value_type & __x) [inline]`

Attempts to insert a std::pair into the unordered_map.

Parameters

<code>__x</code>	Pair to be inserted (see std::make_pair for easy creation of pairs).
------------------	--

Returns

A pair, of which the first element is an iterator that points to the possibly inserted pair, and the second is a bool that is true if the pair was actually inserted.

This function attempts to insert a (key, value) pair into the unordered_map. An unordered_map relies on unique keys and thus a pair is only inserted if its first element (the key) is not already present in the unordered_map.

Insertion requires amortized constant time.

Definition at line 549 of file unordered_map.h.

5.1025.4.32 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type> std::pair<iterator, bool> std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::insert (_Pair && __x) [inline]`

Attempts to insert a std::pair into the unordered_map.

Parameters

<code>__x</code>	Pair to be inserted (see std::make_pair for easy creation of pairs).
------------------	--

Returns

A pair, of which the first element is an iterator that points to the possibly inserted pair, and the second is a bool that is true if the pair was actually inserted.

This function attempts to insert a (key, value) pair into the unordered_map. An unordered_map relies on unique keys and thus a pair is only inserted if its first element (the key) is not already present in the unordered_map.

Insertion requires amortized constant time.

Definition at line 556 of file unordered_map.h.

5.1025.4.33 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> iterator std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::insert (const_iterator __hint, const value_type & __x) [inline]`

Attempts to insert a std::pair into the unordered_map.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__x</code>	Pair to be inserted (see <code>std::make_pair</code> for easy creation of pairs).

Returns

An iterator that points to the element with key of `__x` (may or may not be the pair passed in).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `insert()` does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.↔associative.insert_hints for more on *hinting*.

Insertion requires amortized constant time.

Definition at line 583 of file `unordered_map.h`.

```
5.1025.4.34  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class
               _Alloc = std::allocator<std::pair<const _Key, _Tp>>> template<typename _Pair, typename = typename
               std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type> iterator std::unordered_map<
               _Key, _Tp, _Hash, _Pred, _Alloc>::insert ( const_iterator __hint, _Pair && __x ) [inline]
```

Attempts to insert a `std::pair` into the `unordered_map`.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__x</code>	Pair to be inserted (see <code>std::make_pair</code> for easy creation of pairs).

Returns

An iterator that points to the element with key of `__x` (may or may not be the pair passed in).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `insert()` does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.↔associative.insert_hints for more on *hinting*.

Insertion requires amortized constant time.

Definition at line 590 of file `unordered_map.h`.

```
5.1025.4.35  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
               std::allocator<std::pair<const _Key, _Tp>>> template<typename _InputIterator> void std::unordered_map<
               _Key, _Tp, _Hash, _Pred, _Alloc>::insert ( _InputIterator __first, _InputIterator __last ) [inline]
```

A template function that attempts to insert a range of elements.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be inserted.
<code>__last</code>	Iterator pointing to the end of the range.

Complexity similar to that of the range constructor.

Definition at line 605 of file unordered_map.h.

```
5.1025.4.36  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
              std::allocator<std::pair<const _Key, _Tp>>> void std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc
              >::insert( initializer_list<value_type> &__l ) [inline]
```

Attempts to insert a list of elements into the unordered_map.

Parameters

<code>__l</code>	A std::initializer_list<value_type> of elements to be inserted.
------------------	---

Complexity similar to that of the range constructor.

Definition at line 616 of file unordered_map.h.

```
5.1025.4.37  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
              std::allocator<std::pair<const _Key, _Tp>>> key_equal std::unordered_map<_Key, _Tp, _Hash, _Pred,
              _Alloc>::key_eq( ) const [inline]
```

Returns the key comparison object with which the unordered_map was constructed.

Definition at line 832 of file unordered_map.h.

```
5.1025.4.38  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
              std::allocator<std::pair<const _Key, _Tp>>> float std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc
              >::load_factor( ) const [inline], [noexcept]
```

Returns the average number of elements per bucket.

Definition at line 1014 of file unordered_map.h.

```
5.1025.4.39  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
              std::allocator<std::pair<const _Key, _Tp>>> size_type std::unordered_map<_Key, _Tp, _Hash, _Pred,
              _Alloc>::max_bucket_count( ) const [inline], [noexcept]
```

Returns the maximum number of buckets of the unordered_map.

Definition at line 937 of file unordered_map.h.


```
5.1025.4.40  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> float std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc
>::max_load_factor( ) const    [inline], [noexcept]
```

Returns a positive number that the unordered_map tries to keep the load factor less than or equal to.

Definition at line 1020 of file unordered_map.h.

```
5.1025.4.41  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> void std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc
>::max_load_factor( float _z )    [inline]
```

Change the unordered_map maximum load factor.

Parameters

<code>_z</code>	The new maximum load factor.
-----------------	------------------------------

Definition at line 1028 of file unordered_map.h.

```
5.1025.4.42  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> size_type std::unordered_map<_Key, _Tp, _Hash, _Pred,
_Alloc>::max_size( ) const    [inline], [noexcept]
```

Returns the maximum size of the unordered_map.

Definition at line 306 of file unordered_map.h.

```
5.1025.4.43  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> unordered_map& std::unordered_map<_Key, _Tp, _Hash,
_Pred, _Alloc>::operator=( const unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc> & )    [default]
```

Copy assignment operator.

Referenced by std::unordered_map<_Key, _Tp, _Hash, _Pred>::unordered_map(), and std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::unordered_multimap().

```
5.1025.4.44  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> unordered_map& std::unordered_map<_Key, _Tp, _Hash,
_Pred, _Alloc>::operator=( unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc> && )    [default]
```

Move assignment operator.

```
5.1025.4.45  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> unordered_map& std::unordered_map<_Key, _Tp, _Hash,
_Pred, _Alloc>::operator=( initializer_list<value_type> __l )    [inline]
```

Unordered_map list assignment operator.

Parameters

\leftrightarrow	An initializer_list.
$_ \leftrightarrow$	
\leftrightarrow	
$_ \leftrightarrow$	
$/$	

This function fills an unordered_map with copies of the elements in the initializer list $_ /$.

Note that the assignment completely changes the unordered_map and that the resulting unordered_map's size is the same as the number of elements assigned. Old data may be lost.

Definition at line 280 of file unordered_map.h.

```
5.1025.4.46  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
              std::allocator<std::pair<const _Key, _Tp>>> mapped_type& std::unordered_map<_Key, _Tp, _Hash, _Pred,
              _Alloc>::operator[] ( const key_type & __k ) [inline]
```

Subscript ($[]$) access to unordered_map data.

Parameters

$_ \leftrightarrow$	The key for which data should be retrieved.
$_ k$	

Returns

A reference to the data of the (key,data) pair.

Allows for easy lookup with the subscript ($[]$) operator. Returns data associated with the key specified in subscript. If the key does not exist, a pair with that key is created using default values, which is then returned.

Lookup requires constant time.

Definition at line 903 of file unordered_map.h.

```
5.1025.4.47  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
              std::allocator<std::pair<const _Key, _Tp>>> mapped_type& std::unordered_map<_Key, _Tp, _Hash, _Pred,
              _Alloc>::operator[] ( key_type && __k ) [inline]
```

Subscript ($[]$) access to unordered_map data.

Parameters

$_ \leftrightarrow$	The key for which data should be retrieved.
$_ k$	

Returns

A reference to the data of the (key,data) pair.

Allows for easy lookup with the subscript (`[]`) operator. Returns data associated with the key specified in subscript. If the key does not exist, a pair with that key is created using default values, which is then returned.

Lookup requires constant time.

Definition at line 907 of file `unordered_map.h`.

```
5.1025.4.48  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
              std::allocator<std::pair<const _Key, _Tp>>> void std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc
              >::rehash( size_type __n ) [inline]
```

May rehash the `unordered_map`.

Parameters

<code>__n</code>	The new number of buckets.
------------------	----------------------------

Rehash will occur only if the new number of buckets respect the `unordered_map` maximum load factor.

Definition at line 1039 of file `unordered_map.h`.

```
5.1025.4.49  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
              std::allocator<std::pair<const _Key, _Tp>>> void std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc
              >::reserve( size_type __n ) [inline]
```

Prepare the `unordered_map` for a specified number of elements.

Parameters

<code>__n</code>	Number of elements required.
------------------	------------------------------

Same as `rehash(ceil(n / max_load_factor()))`.

Definition at line 1050 of file `unordered_map.h`.

```
5.1025.4.50  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
              std::allocator<std::pair<const _Key, _Tp>>> size_type std::unordered_map<_Key, _Tp, _Hash, _Pred,
              _Alloc>::size( ) const [inline], [noexcept]
```

Returns the size of the `unordered_map`.

Definition at line 301 of file `unordered_map.h`.

```
5.1025.4.51 template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp> >> void std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc
>::swap ( unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc > & _x ) [inline], [noexcept]
```

Swaps data with another unordered_map.

Parameters

<code>_↔</code>	An unordered_map of the same element and allocator types.
<code>_x</code>	

This exchanges the elements between two unordered_map in constant time. Note that the global std::swap() function is specialized such that std::swap(m1,m2) will feed to this function.

Definition at line 817 of file unordered_map.h.

Referenced by std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::reserve().

The documentation for this class was generated from the following file:

- [unordered_map.h](#)

5.1026 std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc > Class Template Reference

Public Types

- typedef _Hashtable::key_type [key_type](#)
- typedef _Hashtable::value_type [value_type](#)
- typedef _Hashtable::mapped_type [mapped_type](#)
- typedef _Hashtable::hasher [hasher](#)
- typedef _Hashtable::key_equal [key_equal](#)
- typedef _Hashtable::allocator_type [allocator_type](#)
- typedef _Hashtable::pointer [pointer](#)
- typedef _Hashtable::const_pointer [const_pointer](#)
- typedef _Hashtable::reference [reference](#)
- typedef _Hashtable::const_reference [const_reference](#)
- typedef _Hashtable::iterator [iterator](#)
- typedef _Hashtable::const_iterator [const_iterator](#)
- typedef _Hashtable::local_iterator [local_iterator](#)
- typedef _Hashtable::const_local_iterator [const_local_iterator](#)
- typedef _Hashtable::size_type [size_type](#)
- typedef _Hashtable::difference_type [difference_type](#)

Public Member Functions

- `unordered_multimap` ()=default
- `unordered_multimap` (`size_type` __n, const `hasher` &__hf=`hasher`(), const `key_equal` &__eq=`key_equal`(), const `allocator_type` &__a=`allocator_type`())
- `template<typename _InputIterator >`
`unordered_multimap` (_InputIterator __first, _InputIterator __last, `size_type` __n=0, const `hasher` &__hf=`hasher`(), const `key_equal` &__eq=`key_equal`(), const `allocator_type` &__a=`allocator_type`())
- `unordered_multimap` (const `unordered_multimap` &)=default
- `unordered_multimap` (`unordered_multimap` &&)=default
- `unordered_multimap` (const `allocator_type` &__a)
- `unordered_multimap` (const `unordered_multimap` &__ummap, const `allocator_type` &__a)
- `unordered_multimap` (`unordered_multimap` &&__ummap, const `allocator_type` &__a)
- `unordered_multimap` (`initializer_list`< `value_type` > __l, `size_type` __n=0, const `hasher` &__hf=`hasher`(), const `key_equal` &__eq=`key_equal`(), const `allocator_type` &__a=`allocator_type`())
- `unordered_multimap` (`size_type` __n, const `allocator_type` &__a)
- `unordered_multimap` (`size_type` __n, const `hasher` &__hf, const `allocator_type` &__a)
- `template<typename _InputIterator >`
`unordered_multimap` (_InputIterator __first, _InputIterator __last, `size_type` __n, const `allocator_type` &__a)
- `template<typename _InputIterator >`
`unordered_multimap` (_InputIterator __first, _InputIterator __last, `size_type` __n, const `hasher` &__hf, const `allocator_type` &__a)
- `unordered_multimap` (`initializer_list`< `value_type` > __l, `size_type` __n, const `allocator_type` &__a)
- `unordered_multimap` (`initializer_list`< `value_type` > __l, `size_type` __n, const `hasher` &__hf, const `allocator_type` &__a)
- `iterator begin` () noexcept
- `local_iterator begin` (`size_type` __n)
- `size_type bucket` (const `key_type` &__key) const
- `size_type bucket_count` () const noexcept
- `size_type bucket_size` (`size_type` __n) const
- `void clear` () noexcept
- `size_type count` (const `key_type` &__x) const
- `template<typename... _Args>`
`iterator emplace` (_Args &&...__args)
- `template<typename... _Args>`
`iterator emplace_hint` (const `iterator` __pos, _Args &&...__args)
- `bool empty` () const noexcept
- `iterator end` () noexcept
- `local_iterator end` (`size_type` __n)
- `size_type erase` (const `key_type` &__x)
- `iterator erase` (const `iterator` __first, const `iterator` __last)
- `allocator_type get_allocator` () const noexcept
- `hasher hash_function` () const
- `template<typename _InputIterator >`
`void insert` (_InputIterator __first, _InputIterator __last)
- `void insert` (`initializer_list`< `value_type` > __l)
- `key_equal key_eq` () const
- `float load_factor` () const noexcept
- `size_type max_bucket_count` () const noexcept
- `float max_load_factor` () const noexcept
- `void max_load_factor` (float __z)

- `size_type max_size ()` const noexcept
- `unordered_multimap & operator= (const unordered_multimap &)=default`
- `unordered_multimap & operator= (unordered_multimap &&)=default`
- `unordered_multimap & operator= (initializer_list< value_type > __l)`
- `void rehash (size_type __n)`
- `void reserve (size_type __n)`
- `size_type size ()` const noexcept
- `void swap (unordered_multimap &__x)` noexcept(noexcept(_M_h.swap(__x._M_h)))
- `const_iterator begin ()` const noexcept
- `const_iterator cbegin ()` const noexcept
- `const_iterator end ()` const noexcept
- `const_iterator cend ()` const noexcept
- `iterator insert (const value_type &__x)`
- `template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>
iterator insert (_Pair &&__x)`
- `iterator insert (const_iterator __hint, const value_type &__x)`
- `template<typename _Pair, typename = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type>
iterator insert (const_iterator __hint, _Pair &&__x)`
- `iterator erase (const_iterator __position)`
- `iterator erase (iterator __position)`
- `iterator find (const key_type &__x)`
- `const_iterator find (const key_type &__x)` const
- `std::pair< iterator, iterator > equal_range (const key_type &__x)`
- `std::pair< const_iterator, const_iterator > equal_range (const key_type &__x)` const
- `const_local_iterator begin (size_type __n)` const
- `const_local_iterator cbegin (size_type __n)` const
- `const_local_iterator end (size_type __n)` const
- `const_local_iterator cend (size_type __n)` const

Friends

- `template<typename _Key1, typename _Tp1, typename _Hash1, typename _Pred1, typename _Alloc1 >
bool operator== (const unordered_multimap< _Key1, _Tp1, _Hash1, _Pred1, _Alloc1 > &, const unordered_↵
multimap< _Key1, _Tp1, _Hash1, _Pred1, _Alloc1 > &)`

5.1026.1 Detailed Description

```
template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::↵  
::allocator<std::pair<const _Key, _Tp> >>  
class std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >
```

A standard container composed of equivalent keys (possibly containing multiple of each key value) that associates values of another type with the keys.

Template Parameters

<code>_Key</code>	Type of key objects.
<code>_Tp</code>	Type of mapped objects.
<code>_Hash</code>	Hashing function object type, defaults to <code>hash<_Value></code> .
<code>_Pred</code>	Predicate function object type, defaults to <code>equal_to<_Value></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>std::allocator<std::pair<const _Key, _Tp>></code> .

Meets the requirements of a `container`, and `unordered associative container`

The resulting value type of the container is `std::pair<const _Key, _Tp>`.

Base is `_Hashtable`, dispatched at compile time via template alias `__ummap_hashtable`.

Definition at line 1087 of file `unordered_map.h`.

5.1026.2 Member Typedef Documentation

5.1026.2.1 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::allocator_type std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::allocator_type`

Public typedefs.

Definition at line 1101 of file `unordered_map.h`.

5.1026.2.2 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::const_iterator std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::const_iterator`

Iterator-related typedefs.

Definition at line 1111 of file `unordered_map.h`.

5.1026.2.3 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::const_local_iterator std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::const_local_iterator`

Iterator-related typedefs.

Definition at line 1113 of file `unordered_map.h`.

5.1026.2.4 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::const_pointer std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::const_pointer`

Iterator-related typedefs.

Definition at line 1107 of file `unordered_map.h`.

```
5.1026.2.5  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =  
            std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::const_reference std::unordered_multimap<  
            _Key, _Tp, _Hash, _Pred, _Alloc>::const_reference
```

Iterator-related typedefs.

Definition at line 1109 of file unordered_map.h.

```
5.1026.2.6  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =  
            std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::difference_type std::unordered_multimap<  
            _Key, _Tp, _Hash, _Pred, _Alloc>::difference_type
```

Iterator-related typedefs.

Definition at line 1115 of file unordered_map.h.

```
5.1026.2.7  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =  
            std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::hasher std::unordered_multimap<_Key, _Tp,  
            _Hash, _Pred, _Alloc>::hasher
```

Public typedefs.

Definition at line 1099 of file unordered_map.h.

```
5.1026.2.8  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =  
            std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::iterator std::unordered_multimap<_Key,  
            _Tp, _Hash, _Pred, _Alloc>::iterator
```

Iterator-related typedefs.

Definition at line 1110 of file unordered_map.h.

```
5.1026.2.9  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =  
            std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::key_equal std::unordered_multimap<_Key,  
            _Tp, _Hash, _Pred, _Alloc>::key_equal
```

Public typedefs.

Definition at line 1100 of file unordered_map.h.

```
5.1026.2.10 template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =  
            std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::key_type std::unordered_multimap<_Key,  
            _Tp, _Hash, _Pred, _Alloc>::key_type
```

Public typedefs.

Definition at line 1096 of file unordered_map.h.

5.1026.2.11 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::local_iterator std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::local_iterator`

Iterator-related typedefs.

Definition at line 1112 of file unordered_map.h.

5.1026.2.12 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::mapped_type std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::mapped_type`

Public typedefs.

Definition at line 1098 of file unordered_map.h.

5.1026.2.13 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::pointer std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::pointer`

Iterator-related typedefs.

Definition at line 1106 of file unordered_map.h.

5.1026.2.14 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::reference std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::reference`

Iterator-related typedefs.

Definition at line 1108 of file unordered_map.h.

5.1026.2.15 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::size_type std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::size_type`

Iterator-related typedefs.

Definition at line 1114 of file unordered_map.h.

5.1026.2.16 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> typedef _Hashtable::value_type std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::value_type`

Public typedefs.

Definition at line 1097 of file unordered_map.h.

5.1026.3 Constructor & Destructor Documentation

5.1026.3.1 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::unordered_multimap() [default]`

Default constructor.

5.1026.3.2 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::unordered_multimap(size_type __n, const hasher & __hf = hasher(), const key_equal & __eqf = key_equal(), const allocator_type & __a = allocator_type()) [inline],[explicit]`

Default constructor creates no elements.

Parameters

<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eqf</code>	A key equality functor.
<code>__a</code>	An allocator object.

Definition at line 1131 of file `unordered_map.h`.

5.1026.3.3 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> template<typename _InputIterator> std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::unordered_multimap(_InputIterator __first, _InputIterator __last, size_type __n = 0, const hasher & __hf = hasher(), const key_equal & __eqf = key_equal(), const allocator_type & __a = allocator_type()) [inline]`

Builds an `unordered_multimap` from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eqf</code>	A key equality functor.
<code>__a</code>	An allocator object.

Create an `unordered_multimap` consisting of copies of the elements from `[__first,__last)`. This is linear in `N` (where `N` is `distance(__first,__last)`).

Definition at line 1152 of file `unordered_map.h`.

5.1026.3.4 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::unordered_multimap (const unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc> &) [default]`

Copy constructor.

5.1026.3.5 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::unordered_multimap (unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc> &&) [default]`

Move constructor.

5.1026.3.6 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::unordered_multimap (const allocator_type &__a) [inline],[explicit]`

Creates an unordered_multimap with no elements.

Parameters

<code>__a</code>	An allocator object.
------------------	----------------------

Definition at line 1171 of file unordered_map.h.

5.1026.3.7 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::unordered_multimap (initializer_list<value_type> __l, size_type __n = 0, const hasher &__hf = hasher(), const key_equal &__eqf = key_equal(), const allocator_type &__a = allocator_type()) [inline]`

Builds an unordered_multimap from an initializer_list.

Parameters

<code>__l</code>	An initializer_list.
<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eqf</code>	A key equality functor.
<code>__a</code>	An allocator object.

Create an unordered_multimap consisting of copies of the elements in the list. This is linear in N (where N is `__l.size()`).

Definition at line 1206 of file unordered_map.h.

References `std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::operator=()`.

5.1026.4 Member Function Documentation

5.1026.4.1 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> iterator std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::begin () [inline], [noexcept]`

Returns a read/write iterator that points to the first element in the unordered_multimap.

Definition at line 1305 of file unordered_map.h.

5.1026.4.2 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> const_iterator std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::begin () const [inline], [noexcept]`

Returns a read-only (constant) iterator that points to the first element in the unordered_multimap.

Definition at line 1314 of file unordered_map.h.

5.1026.4.3 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> local_iterator std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::begin (size_type __n) [inline]`

Returns a read/write iterator pointing to the first bucket element.

Parameters

<code>__n</code>	The bucket index.
------------------	-------------------

Returns

A read/write local iterator.

Definition at line 1655 of file unordered_map.h.

5.1026.4.4 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> const_local_iterator std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::begin (size_type __n) const [inline]`

Returns a read-only (constant) iterator pointing to the first bucket element.

Parameters

<code>__n</code>	The bucket index.
------------------	-------------------

Returns

A read-only local iterator.

Definition at line 1666 of file unordered_map.h.

```
5.1026.4.5  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
             std::allocator<std::pair<const _Key, _Tp>>> size_type std::unordered_multimap<_Key, _Tp, _Hash, _Pred,
             _Alloc>::bucket_count( ) const    [inline], [noexcept]
```

Returns the number of buckets of the unordered_multimap.

Definition at line 1622 of file unordered_map.h.

```
5.1026.4.6  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
             std::allocator<std::pair<const _Key, _Tp>>> const_iterator std::unordered_multimap<_Key, _Tp, _Hash,
             _Pred, _Alloc>::cbegin( ) const    [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the unordered_multimap.

Definition at line 1318 of file unordered_map.h.

```
5.1026.4.7  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
             std::allocator<std::pair<const _Key, _Tp>>> const_local_iterator std::unordered_multimap<_Key, _Tp,
             _Hash, _Pred, _Alloc>::cbegin( size_type __n ) const    [inline]
```

Returns a read-only (constant) iterator pointing to the first bucket element.

Parameters

\leftrightarrow	The bucket index.
$_n$	

Returns

A read-only local iterator.

Definition at line 1670 of file unordered_map.h.

```
5.1026.4.8  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
             std::allocator<std::pair<const _Key, _Tp>>> const_iterator std::unordered_multimap<_Key, _Tp, _Hash,
             _Pred, _Alloc>::cend( ) const    [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the unordered_multimap.

Definition at line 1340 of file unordered_map.h.

```
5.1026.4.9  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
             std::allocator<std::pair<const _Key, _Tp>>> const_local_iterator std::unordered_multimap<_Key, _Tp,
             _Hash, _Pred, _Alloc>::cend( size_type __n ) const    [inline]
```

Returns a read-only (constant) iterator pointing to one past the last bucket elements.

Parameters

\leftrightarrow	The bucket index.
n	

Returns

A read-only local iterator.

Definition at line 1696 of file unordered_map.h.

```
5.1026.4.10  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
              std::allocator<std::pair<const _Key, _Tp> >> void std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc
              >::clear ( ) [inline], [noexcept]
```

Erases all elements in an unordered_multimap. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1538 of file unordered_map.h.

```
5.1026.4.11  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
              std::allocator<std::pair<const _Key, _Tp> >> size_type std::unordered_multimap< _Key, _Tp, _Hash, _Pred,
              _Alloc >::count ( const key_type & __x ) const [inline]
```

Finds the number of elements.

Parameters

\leftrightarrow	Key to count.
x	

Returns

Number of elements with specified key.

Definition at line 1599 of file unordered_map.h.

```
5.1026.4.12  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>,
              class _Alloc = std::allocator<std::pair<const _Key, _Tp> >> template<typename... _Args> iterator
              std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::emplace ( _Args &&... __args ) [inline]
```

Attempts to build and insert a std::pair into the unordered_multimap.

Parameters

$args$	Arguments used to generate a new pair instance (see std::piecewise_construct for passing arguments to each part of the pair constructor).
--------	---

Returns

An iterator that points to the inserted pair.

This function attempts to build and insert a (key, value) pair into the `unordered_multimap`.

Insertion requires amortized constant time.

Definition at line 1363 of file `unordered_map.h`.

```
5.1026.4.13  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>,
              class _Alloc = std::allocator<std::pair<const _Key, _Tp> >> template<typename... _Args> iterator
              std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc >::emplace_hint( const_iterator __pos, _Args
              &&... __args ) [inline]
```

Attempts to build and insert a `std::pair` into the `unordered_multimap`.

Parameters

<code>__pos</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__args</code>	Arguments used to generate a new pair instance (see <code>std::piecewise_construct</code> for passing arguments to each part of the pair constructor).

Returns

An iterator that points to the element with key of the `std::pair` built from `__args`.

Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints for more on *hinting*.

Insertion requires amortized constant time.

Definition at line 1390 of file `unordered_map.h`.

```
5.1026.4.14  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
              std::allocator<std::pair<const _Key, _Tp> >> bool std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc
              >::empty( ) const [inline], [noexcept]
```

Returns true if the `unordered_multimap` is empty.

Definition at line 1285 of file `unordered_map.h`.

```
5.1026.4.15  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
              std::allocator<std::pair<const _Key, _Tp> >> iterator std::unordered_multimap<_Key, _Tp, _Hash, _Pred,
              _Alloc >::end( ) [inline], [noexcept]
```

Returns a read/write iterator that points one past the last element in the `unordered_multimap`.

Definition at line 1327 of file `unordered_map.h`.

5.1026.4.16 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> const_iterator std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::end () const` `[inline],[noexcept]`

Returns a read-only (constant) iterator that points one past the last element in the unordered_multimap.

Definition at line 1336 of file unordered_map.h.

5.1026.4.17 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> local_iterator std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::end (size_type __n)` `[inline]`

Returns a read/write iterator pointing to one past the last bucket elements.

Parameters

<code>__↔ __n</code>	The bucket index.
--------------------------	-------------------

Returns

A read/write local iterator.

Definition at line 1681 of file unordered_map.h.

5.1026.4.18 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> const_local_iterator std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::end (size_type __n) const` `[inline]`

Returns a read-only (constant) iterator pointing to one past the last bucket elements.

Parameters

<code>__↔ __n</code>	The bucket index.
--------------------------	-------------------

Returns

A read-only local iterator.

Definition at line 1692 of file unordered_map.h.

5.1026.4.19 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> std::pair<iterator, iterator> std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::equal_range (const key_type & __x)` `[inline]`

Finds a subsequence matching given key.

Parameters

<code>_↔</code>	Key to be located.
<code>_x</code>	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

Definition at line 1610 of file `unordered_map.h`.

```
5.1026.4.20  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class
               _Alloc = std::allocator<std::pair<const _Key, _Tp>>> std::pair<const_iterator, const_iterator>
               std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::equal_range ( const key_type & __x ) const
               [inline]
```

Finds a subsequence matching given key.

Parameters

<code>_↔</code>	Key to be located.
<code>_x</code>	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

Definition at line 1614 of file `unordered_map.h`.

```
5.1026.4.21  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
               std::allocator<std::pair<const _Key, _Tp>>> iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred,
               _Alloc >::erase ( const_iterator __position ) [inline]
```

Erases an element from an `unordered_multimap`.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following `__position` prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from an `unordered_multimap`. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1488 of file `unordered_map.h`.

```
5.1026.4.22 template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> iterator std::unordered_multimap<_Key,_Tp,_Hash,_Pred,
_Alloc>::erase( iterator __position ) [inline]
```

Erases an element from an unordered_multimap.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following `__position` prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from an unordered_multimap. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1493 of file unordered_map.h.

```
5.1026.4.23 template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> size_type std::unordered_multimap<_Key,_Tp,_Hash,_Pred,
_Alloc>::erase( const key_type & __x ) [inline]
```

Erases elements according to the provided key.

Parameters

<code>__x</code>	Key of elements to be erased.
------------------	-------------------------------

Returns

The number of elements erased.

This function erases all the elements located by the given key from an unordered_multimap. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1509 of file unordered_map.h.

```
5.1026.4.24 template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> iterator std::unordered_multimap<_Key,_Tp,_Hash,_Pred,
_Alloc>::erase( const_iterator __first, const_iterator __last ) [inline]
```

Erases a [`__first`,`__last`) range of elements from an unordered_multimap.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be erased.
<code>__last</code>	Iterator pointing to the end of the range to be erased.

Returns

The iterator `__last`.

This function erases a sequence of elements from an `unordered_multimap`. Note that this function only erases the elements, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1528 of file `unordered_map.h`.

```
5.1026.4.25  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
              std::allocator<std::pair<const _Key, _Tp>>> iterator std::unordered_multimap<_Key, _Tp, _Hash, _Pred,
              _Alloc >::find ( const key_type & __x ) [inline]
```

Tries to locate an element in an `unordered_multimap`.

Parameters

<code>__x</code>	Key to be located.
------------------	--------------------

Returns

Iterator pointing to sought-after element, or `end()` if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (`end()`) iterator.

Definition at line 1585 of file `unordered_map.h`.

```
5.1026.4.26  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
              std::allocator<std::pair<const _Key, _Tp>>> const_iterator std::unordered_multimap<_Key, _Tp, _Hash,
              _Pred, _Alloc >::find ( const key_type & __x ) const [inline]
```

Tries to locate an element in an `unordered_multimap`.

Parameters

<code>__x</code>	Key to be located.
------------------	--------------------

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (end()) iterator.

Definition at line 1589 of file unordered_map.h.

```
5.1026.4.27  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
             std::allocator<std::pair<const _Key, _Tp>>> allocator_type std::unordered_multimap< _Key, _Tp, _Hash,
             _Pred, _Alloc >::get_allocator ( ) const    [inline], [noexcept]
```

Returns the allocator object with which the unordered_multimap was constructed.

Definition at line 1278 of file unordered_map.h.

```
5.1026.4.28  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
             std::allocator<std::pair<const _Key, _Tp>>> hasher std::unordered_multimap< _Key, _Tp, _Hash, _Pred,
             _Alloc >::hash_function ( ) const    [inline]
```



Returns the hash functor object with which the unordered_multimap was constructed.

Definition at line 1561 of file unordered_map.h.

```
5.1026.4.29  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
             std::allocator<std::pair<const _Key, _Tp>>> iterator std::unordered_multimap< _Key, _Tp, _Hash, _Pred,
             _Alloc >::insert ( const value_type & __x )    [inline]
```

Inserts a std::pair into the unordered_multimap.

Parameters

	Pair to be inserted (see std::make_pair for easy creation of pairs).
	

Returns

An iterator that points to the inserted pair.

Insertion requires amortized constant time.

Definition at line 1404 of file unordered_map.h.

```
5.1026.4.30  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>,
             class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> template<typename _Pair, typename
             = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type> iterator
             std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::insert ( _Pair && __x )    [inline]
```

Inserts a std::pair into the unordered_multimap.

Parameters

<code>__x</code>	Pair to be inserted (see <code>std::make_pair</code> for easy creation of pairs).
------------------	---

Returns

An iterator that points to the inserted pair.

Insertion requires amortized constant time.

Definition at line 1411 of file `unordered_map.h`.

```
5.1026.4.31  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
              std::allocator<std::pair<const _Key, _Tp>>> iterator std::unordered_multimap<_Key, _Tp, _Hash, _Pred,
              _Alloc>::insert ( const_iterator __hint, const value_type &__x ) [inline]
```

Inserts a `std::pair` into the `unordered_multimap`.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__x</code>	Pair to be inserted (see <code>std::make_pair</code> for easy creation of pairs).

Returns

An iterator that points to the element with key of `__x` (may or may not be the pair passed in).

Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints for more on *hinting*.

Insertion requires amortized constant time.

Definition at line 1436 of file `unordered_map.h`.

```
5.1026.4.32  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>,
              class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> template<typename _Pair, typename
              = typename std::enable_if<std::is_constructible<value_type, _Pair&&>::value>::type> iterator
              std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::insert ( const_iterator __hint, _Pair && __x )
              [inline]
```

Inserts a `std::pair` into the `unordered_multimap`.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the pair should be inserted.
<code>__x</code>	Pair to be inserted (see <code>std::make_pair</code> for easy creation of pairs).

Returns

An iterator that points to the element with key of `__x` (may or may not be the pair passed in).

Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

See https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.↔associative.insert_hints for more on *hinting*.

Insertion requires amortized constant time.

Definition at line 1443 of file `unordered_map.h`.

```
5.1026.4.33  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class
              _Alloc = std::allocator<std::pair<const _Key, _Tp>>> template<typename _InputIterator > void
              std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::insert ( _InputIterator __first, _InputIterator __last
              ) [inline]
```

A template function that attempts to insert a range of elements.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be inserted.
<code>__last</code>	Iterator pointing to the end of the range.

Complexity similar to that of the range constructor.

Definition at line 1458 of file `unordered_map.h`.

```
5.1026.4.34  template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
              std::allocator<std::pair<const _Key, _Tp>>> void std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc
              >::insert ( initializer_list<value_type> __l ) [inline]
```

Attempts to insert a list of elements into the `unordered_multimap`.

Parameters

<code>↵ _↵ ↵ _↵ /</code>	A <code>std::initializer_list<value_type></code> of elements to be inserted.
--	--

Complexity similar to that of the range constructor.

Definition at line 1470 of file unordered_map.h.

```
5.1026.4.35 template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> key_equal std::unordered_multimap< _Key, _Tp, _Hash,
_Pred, _Alloc >::key_eq ( ) const [inline]
```

Returns the key comparison object with which the unordered_multimap was constructed.

Definition at line 1567 of file unordered_map.h.

```
5.1026.4.36 template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> float std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc
>::load_factor ( ) const [inline], [noexcept]
```

Returns the average number of elements per bucket.

Definition at line 1704 of file unordered_map.h.

```
5.1026.4.37 template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> size_type std::unordered_multimap< _Key, _Tp, _Hash, _Pred,
_Alloc >::max_bucket_count ( ) const [inline], [noexcept]
```

Returns the maximum number of buckets of the unordered_multimap.

Definition at line 1627 of file unordered_map.h.

```
5.1026.4.38 template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> float std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc
>::max_load_factor ( ) const [inline], [noexcept]
```

Returns a positive number that the unordered_multimap tries to keep the load factor less than or equal to.

Definition at line 1710 of file unordered_map.h.

```
5.1026.4.39 template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc =
std::allocator<std::pair<const _Key, _Tp>>> void std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc
>::max_load_factor ( float __z ) [inline]
```

Change the unordered_multimap maximum load factor.

Parameters

<code>__z</code>	The new maximum load factor.
------------------	------------------------------

Definition at line 1718 of file unordered_map.h.

5.1026.4.40 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> size_type std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::max_size() const [inline], [noexcept]`

Returns the maximum size of the unordered_multimap.

Definition at line 1295 of file unordered_map.h.

5.1026.4.41 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> unordered_multimap& std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::operator=(const unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc> &) [default]`

Copy assignment operator.

5.1026.4.42 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> unordered_multimap& std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::operator=(unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc> &&) [default]`

Move assignment operator.

5.1026.4.43 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> unordered_multimap& std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::operator=(initializer_list<value_type> __l) [inline]`

Unordered_multimap list assignment operator.

Parameters

↵	An initializer_list.
↵	
↵	
↵	
/	

This function fills an unordered_multimap with copies of the elements in the initializer list __l.

Note that the assignment completely changes the unordered_multimap and that the resulting unordered_multimap's size is the same as the number of elements assigned. Old data may be lost.

Definition at line 1269 of file unordered_map.h.

5.1026.4.44 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> void std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::rehash(size_type __n) [inline]`

May rehash the unordered_multimap.

Parameters

\leftarrow _n	The new number of buckets.
--------------------	----------------------------

Rehash will occur only if the new number of buckets respect the unordered_multimap maximum load factor.

Definition at line 1729 of file unordered_map.h.

5.1026.4.45 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> void std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::reserve (size_type __n) [inline]`

Prepare the unordered_multimap for a specified number of elements.

Parameters

\leftarrow _n	Number of elements required.
--------------------	------------------------------

Same as rehash(ceil(n / max_load_factor())).

Definition at line 1740 of file unordered_map.h.

References std::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>::swap(), and std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::swap().

5.1026.4.46 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> size_type std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::size () const [inline], [noexcept]`

Returns the size of the unordered_multimap.

Definition at line 1290 of file unordered_map.h.

5.1026.4.47 `template<class _Key, class _Tp, class _Hash = hash<_Key>, class _Pred = std::equal_to<_Key>, class _Alloc = std::allocator<std::pair<const _Key, _Tp>>> void std::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>::swap (unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc> &__x) [inline], [noexcept]`

Swaps data with another unordered_multimap.

Parameters

\leftarrow _x	An unordered_multimap of the same element and allocator types.
--------------------	--

This exchanges the elements between two unordered_multimap in constant time. Note that the global std::swap() function is specialized such that std::swap(m1,m2) will feed to this function.

Definition at line 1552 of file unordered_map.h.

Referenced by std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >::reserve().

The documentation for this class was generated from the following file:

- [unordered_map.h](#)

5.1027 std::unordered_multiset< _Value, _Hash, _Pred, _Alloc > Class Template Reference

Public Types

- typedef _Hashtable::key_type [key_type](#)
- typedef _Hashtable::value_type [value_type](#)
- typedef _Hashtable::hasher [hasher](#)
- typedef _Hashtable::key_equal [key_equal](#)
- typedef _Hashtable::allocator_type [allocator_type](#)
- typedef _Hashtable::pointer [pointer](#)
- typedef _Hashtable::const_pointer [const_pointer](#)
- typedef _Hashtable::reference [reference](#)
- typedef _Hashtable::const_reference [const_reference](#)
- typedef _Hashtable::iterator [iterator](#)
- typedef _Hashtable::const_iterator [const_iterator](#)
- typedef _Hashtable::local_iterator [local_iterator](#)
- typedef _Hashtable::const_local_iterator [const_local_iterator](#)
- typedef _Hashtable::size_type [size_type](#)
- typedef _Hashtable::difference_type [difference_type](#)

Public Member Functions

- [unordered_multiset](#) ()=default
- [unordered_multiset](#) (size_type __n, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- template<typename _InputIterator >
[unordered_multiset](#) (_InputIterator __first, _InputIterator __last, size_type __n=0, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- [unordered_multiset](#) (const unordered_multiset &)=default
- [unordered_multiset](#) (unordered_multiset &&)=default
- [unordered_multiset](#) (initializer_list< value_type > __l, size_type __n=0, const hasher &__hf=hasher(), const key_equal &__eq=key_equal(), const allocator_type &__a=allocator_type())
- [unordered_multiset](#) (const allocator_type &__a)
- [unordered_multiset](#) (const unordered_multiset &__umset, const allocator_type &__a)
- [unordered_multiset](#) (unordered_multiset &&__umset, const allocator_type &__a)
- [unordered_multiset](#) (size_type __n, const allocator_type &__a)
- [unordered_multiset](#) (size_type __n, const hasher &__hf, const allocator_type &__a)
- template<typename _InputIterator >
[unordered_multiset](#) (_InputIterator __first, _InputIterator __last, size_type __n, const allocator_type &__a)

- `template<typename _InputIterator >`
`unordered_multiset` (`_InputIterator` __first, `_InputIterator` __last, `size_type` __n, const `hasher` &__hf, const `allocator_type` &__a)
 - `unordered_multiset` (`initializer_list`< `value_type` > __l, `size_type` __n, const `allocator_type` &__a)
 - `unordered_multiset` (`initializer_list`< `value_type` > __l, `size_type` __n, const `hasher` &__hf, const `allocator_type` &__a)
 - `size_type` `bucket` (const `key_type` &__key) const
 - `size_type` `bucket_count` () const noexcept
 - `size_type` `bucket_size` (`size_type` __n) const
 - `const_iterator` `cbegin` () const noexcept
 - `const_iterator` `cend` () const noexcept
 - void `clear` () noexcept
 - `size_type` `count` (const `key_type` &__x) const
 - `template<typename... _Args>`
`iterator` `emplace` (`_Args` &&... __args)
 - `template<typename... _Args>`
`iterator` `emplace_hint` (const `iterator` __pos, `_Args` &&... __args)
 - bool `empty` () const noexcept
 - `size_type` `erase` (const `key_type` &__x)
 - `iterator` `erase` (const `iterator` __first, const `iterator` __last)
 - `allocator_type` `get_allocator` () const noexcept
 - `hasher` `hash_function` () const
 - `template<typename _InputIterator >`
void `insert` (`_InputIterator` __first, `_InputIterator` __last)
 - void `insert` (`initializer_list`< `value_type` > __l)
 - `key_equal` `key_eq` () const
 - float `load_factor` () const noexcept
 - `size_type` `max_bucket_count` () const noexcept
 - float `max_load_factor` () const noexcept
 - void `max_load_factor` (float __z)
 - `size_type` `max_size` () const noexcept
 - `unordered_multiset` & `operator=` (const `unordered_multiset` &)=default
 - `unordered_multiset` & `operator=` (`unordered_multiset` &&)=default
 - `unordered_multiset` & `operator=` (`initializer_list`< `value_type` > __l)
 - void `rehash` (`size_type` __n)
 - void `reserve` (`size_type` __n)
 - `size_type` `size` () const noexcept
 - void `swap` (`unordered_multiset` &__x) noexcept(noexcept(__M_h.swap(__x._M_h)))
-
- `iterator` `begin` () noexcept
 - `const_iterator` `begin` () const noexcept
-
- `iterator` `end` () noexcept
 - `const_iterator` `end` () const noexcept
-
- `iterator` `insert` (const `value_type` &__x)
 - `iterator` `insert` (`value_type` &&__x)
-
- `iterator` `insert` (const `iterator` __hint, const `value_type` &__x)

- [iterator insert](#) ([const_iterator](#) __hint, [value_type](#) &&__x)
- [iterator erase](#) ([const_iterator](#) __position)
- [iterator erase](#) ([iterator](#) __position)
- [iterator find](#) ([const key_type](#) &__x)
- [const_iterator find](#) ([const key_type](#) &__x) const
- [std::pair< iterator, iterator > equal_range](#) ([const key_type](#) &__x)
- [std::pair< const_iterator, const_iterator > equal_range](#) ([const key_type](#) &__x) const
- [local_iterator begin](#) ([size_type](#) __n)
- [const_local_iterator begin](#) ([size_type](#) __n) const
- [const_local_iterator cbegin](#) ([size_type](#) __n) const
- [local_iterator end](#) ([size_type](#) __n)
- [const_local_iterator end](#) ([size_type](#) __n) const
- [const_local_iterator cend](#) ([size_type](#) __n) const

Friends

- [template<typename _Value1, typename _Hash1, typename _Pred1, typename _Alloc1 > bool operator==](#) ([const unordered_multiset< _Value1, _Hash1, _Pred1, _Alloc1 >](#) &, [const unordered_multiset< _Value1, _Hash1, _Pred1, _Alloc1 >](#) &)

5.1027.1 Detailed Description

```
template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>>
class std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >
```

A standard container composed of equivalent keys (possibly containing multiple of each key value) in which the elements' keys are the elements themselves.

Template Parameters

_Value	Type of key objects.
_Hash	Hashing function object type, defaults to hash<_Value> .
_Pred	Predicate function object type, defaults to equal_to<_Value> .
_Alloc	Allocator type, defaults to allocator<_Key> .

Meets the requirements of a [container](#), and [unordered associative container](#)

Base is [_Hashtable](#), dispatched at compile time via template alias [__umset_hashtable](#).

Definition at line 767 of file unordered_set.h.

5.1027.2 Member Typedef Documentation

5.1027.2.1 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> typedef _Hashtable::allocator_type std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::allocator_type`

Public typedefs.

Definition at line 780 of file unordered_set.h.

5.1027.2.2 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> typedef _Hashtable::const_iterator std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::const_iterator`

Iterator-related typedefs.

Definition at line 790 of file unordered_set.h.

5.1027.2.3 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> typedef _Hashtable::const_local_iterator std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::const_local_iterator`

Iterator-related typedefs.

Definition at line 792 of file unordered_set.h.

5.1027.2.4 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> typedef _Hashtable::const_pointer std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::const_pointer`

Iterator-related typedefs.

Definition at line 786 of file unordered_set.h.

5.1027.2.5 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> typedef _Hashtable::const_reference std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::const_reference`

Iterator-related typedefs.

Definition at line 788 of file unordered_set.h.

5.1027.2.6 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> typedef _Hashtable::difference_type std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::difference_type`

Iterator-related typedefs.

Definition at line 794 of file unordered_set.h.

```
5.1027.2.7 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> typedef _Hashtable::hasher std::unordered_multiset<_Value, _Hash, _Pred, _Alloc
>::hasher
```

Public typedefs.

Definition at line 778 of file `unordered_set.h`.

```
5.1027.2.8 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> typedef _Hashtable::iterator std::unordered_multiset<_Value, _Hash, _Pred, _Alloc
>::iterator
```

Iterator-related typedefs.

Definition at line 789 of file `unordered_set.h`.

```
5.1027.2.9 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> typedef _Hashtable::key_equal std::unordered_multiset<_Value, _Hash, _Pred, _Alloc
>::key_equal
```

Public typedefs.

Definition at line 779 of file `unordered_set.h`.

```
5.1027.2.10 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> typedef _Hashtable::key_type std::unordered_multiset<_Value, _Hash, _Pred, _Alloc
>::key_type
```

Public typedefs.

Definition at line 776 of file `unordered_set.h`.

```
5.1027.2.11 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> typedef _Hashtable::local_iterator std::unordered_multiset<_Value, _Hash, _Pred,
_Alloc>::local_iterator
```

Iterator-related typedefs.

Definition at line 791 of file `unordered_set.h`.

```
5.1027.2.12 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> typedef _Hashtable::pointer std::unordered_multiset<_Value, _Hash, _Pred, _Alloc
>::pointer
```

Iterator-related typedefs.

Definition at line 785 of file `unordered_set.h`.

5.1027.2.13 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> typedef _Hashtable::reference std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::reference`

Iterator-related typedefs.

Definition at line 787 of file unordered_set.h.

5.1027.2.14 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> typedef _Hashtable::size_type std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::size_type`

Iterator-related typedefs.

Definition at line 793 of file unordered_set.h.

5.1027.2.15 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> typedef _Hashtable::value_type std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::value_type`

Public typedefs.

Definition at line 777 of file unordered_set.h.

5.1027.3 Constructor & Destructor Documentation

5.1027.3.1 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::unordered_multiset () [default]`

Default constructor.

5.1027.3.2 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::unordered_multiset (size_type __n, const hasher & __hf = hasher(), const key_equal & __eqf = key_equal(), const allocator_type & __a = allocator_type()) [inline],[explicit]`

Default constructor creates no elements.

Parameters

<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eqf</code>	A key equality functor.
<code>__a</code>	An allocator object.

Definition at line 810 of file unordered_set.h.

```
5.1027.3.3 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> template<typename _InputIterator > std::unordered_multiset< _Value, _Hash, _Pred,
_Alloc >::unordered_multiset ( _InputIterator __first, _InputIterator __last, size_type __n = 0, const hasher &
__hf = hasher(), const key_equal & __eqf = key_equal(), const allocator_type & __a = allocator_type() )
[inline]
```

Builds an unordered_multiset from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eqf</code>	A key equality functor.
<code>__a</code>	An allocator object.

Create an unordered_multiset consisting of copies of the elements from [`__first`,`__last`). This is linear in N (where N is distance(`__first`,`__last`)).

Definition at line 831 of file unordered_set.h.

```
5.1027.3.4 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::unordered_multiset (
const unordered_multiset< _Value, _Hash, _Pred, _Alloc > & ) [default]
```

Copy constructor.

```
5.1027.3.5 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::unordered_multiset (
unordered_multiset< _Value, _Hash, _Pred, _Alloc > && ) [default]
```

Move constructor.

```
5.1027.3.6 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::unordered_multiset (
initializer_list< value_type > __l, size_type __n = 0, const hasher & __hf = hasher(), const key_equal &
__eqf = key_equal(), const allocator_type & __a = allocator_type() ) [inline]
```

Builds an unordered_multiset from an initializer_list.

Parameters

<code>__l</code>	An initializer_list.
<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eqf</code>	A key equality functor.
<code>__a</code>	An allocator object.

Create an `unordered_multiset` consisting of copies of the elements in the list. This is linear in N (where N is `__l.size()`).

Definition at line 856 of file `unordered_set.h`.

References `std::unordered_set<_Value, _Hash, _Pred, _Alloc>::operator=()`.

```
5.1027.3.7 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::unordered_multiset (
const allocator_type &__a ) [inline], [explicit]
```

Creates an `unordered_multiset` with no elements.

Parameters

<code>__a</code>	An allocator object.
------------------	----------------------

Definition at line 877 of file `unordered_set.h`.

5.1027.4 Member Function Documentation

```
5.1027.4.1 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> iterator std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::begin ( )
[inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the `unordered_multiset`.

Definition at line 985 of file `unordered_set.h`.

```
5.1027.4.2 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> const_iterator std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::begin ( )
const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the `unordered_multiset`.

Definition at line 989 of file `unordered_set.h`.

```
5.1027.4.3 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> local_iterator std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::begin (
size_type __n ) [inline]
```

Returns a read-only (constant) iterator pointing to the first bucket element.

Parameters

<code>__n</code>	The bucket index.
------------------	-------------------

Returns

A read-only local iterator.

Definition at line 1313 of file unordered_set.h.

```
5.1027.4.4  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
            std::allocator<_Value>> const_local_iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc
            >::begin ( size_type __n ) const    [inline]
```

Returns a read-only (constant) iterator pointing to the first bucket element.

Parameters

\leftrightarrow	The bucket index.
n	

Returns

A read-only local iterator.

Definition at line 1317 of file unordered_set.h.

```
5.1027.4.5  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
            std::allocator<_Value>> size_type std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::bucket_count (
            ) const    [inline], [noexcept]
```

Returns the number of buckets of the unordered_multiset.

Definition at line 1279 of file unordered_set.h.

```
5.1027.4.6  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
            std::allocator<_Value>> const_iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::cbegin ( )
            const    [inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the unordered_multiset.

Definition at line 1012 of file unordered_set.h.

```
5.1027.4.7  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
            std::allocator<_Value>> const_local_iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc
            >::cbegin ( size_type __n ) const    [inline]
```

Returns a read-only (constant) iterator pointing to the first bucket element.

Parameters

\leftrightarrow	The bucket index.
n	

Returns

A read-only local iterator.

Definition at line 1321 of file unordered_set.h.

```
5.1027.4.8  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
             std::allocator<_Value>> const_iterator std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::cend ( )
             const [inline],[noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the unordered_multiset.

Definition at line 1020 of file unordered_set.h.

```
5.1027.4.9  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
             std::allocator<_Value>> const_local_iterator std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::cend
             ( size_type __n ) const [inline]
```

Returns a read-only (constant) iterator pointing to one past the last bucket elements.

Parameters

$_n$	The bucket index.
-------	-------------------

Returns

A read-only local iterator.

Definition at line 1341 of file unordered_set.h.

```
5.1027.4.10 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc
              = std::allocator<_Value>> void std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::clear ( )
              [inline],[noexcept]
```

Erases all elements in an unordered_multiset.

Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1196 of file unordered_set.h.

```
5.1027.4.11 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
             std::allocator<_Value>> size_type std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::count ( const
             key_type &__x ) const [inline]
```

Finds the number of elements.

Parameters

<code>_↔</code>	Element to located.
<code>_X</code>	

Returns

Number of elements with specified key.

Definition at line 1256 of file unordered_set.h.

```
5.1027.4.12 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> template<typename... _Args> iterator std::unordered_multiset< _Value, _Hash,
_Pred, _Alloc >::emplace ( _Args &&... __args ) [inline]
```

Builds and insert an element into the unordered_multiset.

Parameters

<code>__args</code>	Arguments used to generate an element.
---------------------	--

Returns

An iterator that points to the inserted element.

Insertion requires amortized constant time.

Definition at line 1034 of file unordered_set.h.

```
5.1027.4.13 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> template<typename... _Args> iterator std::unordered_multiset< _Value, _Hash,
_Pred, _Alloc >::emplace_hint ( const_iterator __pos, _Args &&... __args ) [inline]
```

Inserts an element into the unordered_multiset.

Parameters

<code>__pos</code>	An iterator that serves as a hint as to where the element should be inserted.
<code>__args</code>	Arguments used to generate the element to be inserted.

Returns

An iterator that points to the inserted element.

Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.↵html#containers.associative.insert_hints

Insertion requires amortized constant time.

Definition at line 1056 of file unordered_set.h.

```
5.1027.4.14  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> bool std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::empty ( ) const
              [inline], [noexcept]
```

Returns true if the unordered_multiset is empty.

Definition at line 964 of file unordered_set.h.

```
5.1027.4.15  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::end ( )
              [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the unordered_multiset.

Definition at line 999 of file unordered_set.h.

```
5.1027.4.16  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> const_iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::end ( )
              const [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the unordered_multiset.

Definition at line 1003 of file unordered_set.h.

```
5.1027.4.17  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> local_iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::end (
              size_type __n ) [inline]
```

Returns a read-only (constant) iterator pointing to one past the last bucket elements.

Parameters

<code>↵</code>	The bucket index.
<code>__n</code>	

Returns

A read-only local iterator.

Definition at line 1333 of file unordered_set.h.

5.1027.4.18 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> const_local_iterator std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::end (size_type __n) const [inline]`

Returns a read-only (constant) iterator pointing to one past the last bucket elements.

Parameters

<code>__n</code>	The bucket index.
------------------	-------------------

Returns

A read-only local iterator.

Definition at line 1337 of file unordered_set.h.

5.1027.4.19 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> std::pair<iterator, iterator> std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::equal_range (const key_type & __x) [inline]`

Finds a subsequence matching given key.

Parameters

<code>__x</code>	Key to be located.
------------------	--------------------

Returns

Pair of iterators that possibly points to the subsequence matching given key.

Definition at line 1267 of file unordered_set.h.

5.1027.4.20 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> std::pair<const_iterator, const_iterator> std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::equal_range (const key_type & __x) const [inline]`

Finds a subsequence matching given key.

Parameters

<code>__x</code>	Key to be located.
------------------	--------------------

Returns

Pair of iterators that possibly points to the subsequence matching given key.

Definition at line 1271 of file unordered_set.h.

```
5.1027.4.21  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> iterator std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::erase (
              const_iterator __position ) [inline]
```

Erases an element from an unordered_multiset.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following `__position` prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from an unordered_multiset.

Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1142 of file unordered_set.h.

```
5.1027.4.22  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> iterator std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::erase ( iterator
              __position ) [inline]
```

Erases an element from an unordered_multiset.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following `__position` prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from an unordered_multiset.

Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1147 of file unordered_set.h.

5.1027.4.23 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> size_type std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::erase (const key_type &_x) [inline]`

Erases elements according to the provided key.

Parameters

<code>_x</code>	Key of element to be erased.
-----------------	------------------------------

Returns

The number of elements erased.

This function erases all the elements located by the given key from an `unordered_multiset`.

Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1165 of file `unordered_set.h`.

5.1027.4.24 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> iterator std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::erase (const_iterator __first, const_iterator __last) [inline]`

Erases a [`__first`,`__last`) range of elements from an `unordered_multiset`.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be erased.
<code>__last</code>	Iterator pointing to the end of the range to be erased.

Returns

The iterator `__last`.

This function erases a sequence of elements from an `unordered_multiset`.

Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1185 of file `unordered_set.h`.

5.1027.4.25 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> iterator std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::find (const key_type &_x) [inline]`

Tries to locate an element in an `unordered_multiset`.

Parameters

<code>_↔</code>	Element to be located.
<code>_x</code>	

Returns

Iterator pointing to sought-after element, or `end()` if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (`end()`) iterator.

Definition at line 1242 of file `unordered_set.h`.

```
5.1027.4.26  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> const_iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::find (
              const key_type & _x ) const    [inline]
```

Tries to locate an element in an `unordered_multiset`.

Parameters

<code>_↔</code>	Element to be located.
<code>_x</code>	

Returns

Iterator pointing to sought-after element, or `end()` if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (`end()`) iterator.

Definition at line 1246 of file `unordered_set.h`.

```
5.1027.4.27  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc
              = std::allocator<_Value>> allocator_type std::unordered_multiset< _Value, _Hash, _Pred, _Alloc
              >::get_allocator ( ) const    [inline], [noexcept]
```

Returns the allocator object with which the `unordered_multiset` was constructed.

Definition at line 957 of file `unordered_set.h`.

```
5.1027.4.28  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> hasher std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::hash_function ( )
              const    [inline]
```

Returns the hash functor object with which the `unordered_multiset` was constructed.

Definition at line 1218 of file `unordered_set.h`.

5.1027.4.29 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> iterator std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::insert (const value_type &__x) [inline]`

Inserts an element into the `unordered_multiset`.

Parameters

<code>_↔</code>	Element to be inserted.
<code>_x</code>	

Returns

An iterator that points to the inserted element.

Insertion requires amortized constant time.

Definition at line 1068 of file `unordered_set.h`.

```
5.1027.4.30  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::insert (
              value_type && _x ) [inline]
```

Inserts an element into the `unordered_multiset`.

Parameters

<code>_↔</code>	Element to be inserted.
<code>_x</code>	

Returns

An iterator that points to the inserted element.

Insertion requires amortized constant time.

Definition at line 1072 of file `unordered_set.h`.

```
5.1027.4.31  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> iterator std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::insert (
              const_iterator __hint, const value_type & _x ) [inline]
```

Inserts an element into the `unordered_multiset`.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the element should be inserted.
<code>__x</code>	Element to be inserted.

Returns

An iterator that points to the inserted element.

Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.↵html#containers.associative.insert_hints

Insertion requires amortized constant.

Definition at line 1094 of file unordered_set.h.

5.1027.4.32 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> iterator std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::insert (const_iterator __hint, value_type && __x) [inline]`

Inserts an element into the unordered_multiset.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the element should be inserted.
<code>__x</code>	Element to be inserted.

Returns

An iterator that points to the inserted element.

Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.↵html#containers.associative.insert_hints

Insertion requires amortized constant.

Definition at line 1098 of file unordered_set.h.

5.1027.4.33 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> template<typename _InputIterator > void std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::insert (_InputIterator __first, _InputIterator __last) [inline]`

A template function that inserts a range of elements.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be inserted.
<code>__last</code>	Iterator pointing to the end of the range.

Complexity similar to that of the range constructor.

Definition at line 1112 of file unordered_set.h.

5.1027.4.34 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> void std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::insert (initializer_list<value_type> &_l) [inline]`

Inserts a list of elements into the unordered_multiset.

Parameters

↩	A std::initializer_list<value_type> of elements to be inserted.
_↩	
↩	
_↩	
/	

Complexity similar to that of the range constructor.

Definition at line 1123 of file unordered_set.h.

5.1027.4.35 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> key_equal std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::key_eq () const [inline]`

Returns the key comparison object with which the unordered_multiset was constructed.

Definition at line 1224 of file unordered_set.h.

5.1027.4.36 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> float std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::load_factor () const [inline], [noexcept]`

Returns the average number of elements per bucket.

Definition at line 1349 of file unordered_set.h.

5.1027.4.37 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> size_type std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::max_bucket_count () const [inline], [noexcept]`

Returns the maximum number of buckets of the unordered_multiset.

Definition at line 1284 of file unordered_set.h.

5.1027.4.38 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> float std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::max_load_factor () const [inline], [noexcept]`

Returns a positive number that the unordered_multiset tries to keep the load factor less than or equal to.

Definition at line 1355 of file unordered_set.h.

5.1027.4.39 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> void std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::max_load_factor (float _z) [inline]`

Change the unordered_multiset maximum load factor.

Parameters

<code>_↔</code>	The new maximum load factor.
<code>_Z</code>	

Definition at line 1363 of file unordered_set.h.

```
5.1027.4.40  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> size_type std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::max_size ( )
              const [inline], [noexcept]
```

Returns the maximum size of the unordered_multiset.

Definition at line 974 of file unordered_set.h.

```
5.1027.4.41  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> unordered_multiset& std::unordered_multiset< _Value, _Hash, _Pred, _Alloc
              >::operator= ( const unordered_multiset< _Value, _Hash, _Pred, _Alloc > & ) [default]
```

Copy assignment operator.

```
5.1027.4.42  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> unordered_multiset& std::unordered_multiset< _Value, _Hash, _Pred, _Alloc
              >::operator= ( unordered_multiset< _Value, _Hash, _Pred, _Alloc > && ) [default]
```

Move assignment operator.

```
5.1027.4.43  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> unordered_multiset& std::unordered_multiset< _Value, _Hash, _Pred, _Alloc
              >::operator= ( initializer_list< value_type > __l ) [inline]
```

Unordered_multiset list assignment operator.

Parameters

<code>↔</code>	An initializer_list.
<code>_↔</code>	
<code>↔</code>	
<code>_↔</code>	
<code>/</code>	

This function fills an unordered_multiset with copies of the elements in the initializer list __l.

Note that the assignment completely changes the unordered_multiset and that the resulting unordered_multiset's size is the same as the number of elements assigned. Old data may be lost.

Definition at line 948 of file unordered_set.h.

5.1027.4.44 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> void std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::rehash (size_type __n) [inline]`

May rehash the unordered_multiset.

Parameters

<code>__n</code>	The new number of buckets.
------------------	----------------------------

Rehash will occur only if the new number of buckets respect the unordered_multiset maximum load factor.

Definition at line 1374 of file unordered_set.h.

5.1027.4.45 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> void std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::reserve (size_type __n) [inline]`

Prepare the unordered_multiset for a specified number of elements.

Parameters

<code>__n</code>	Number of elements required.
------------------	------------------------------

Same as rehash(ceil(n / max_load_factor())).

Definition at line 1385 of file unordered_set.h.

References `std::unordered_set<_Value, _Hash, _Pred, _Alloc>::swap()`, and `std::unordered_multiset<_Value, __Hash, _Pred, _Alloc>::swap()`.

5.1027.4.46 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> size_type std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::size () const [inline],[noexcept]`

Returns the size of the unordered_multiset.

Definition at line 969 of file unordered_set.h.

5.1027.4.47 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> void std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::swap (unordered_multiset<_Value, _Hash, _Pred, _Alloc> &__x) [inline],[noexcept]`

Swaps data with another unordered_multiset.

Parameters

<code>_↔_X</code>	An unordered_multiset of the same element and allocator types.
-----------------------------------	--

This exchanges the elements between two sets in constant time. Note that the global `std::swap()` function is specialized such that `std::swap(s1,s2)` will feed to this function.

Definition at line 1209 of file `unordered_set.h`.

Referenced by `std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >::reserve()`.

The documentation for this class was generated from the following file:

- [unordered_set.h](#)

5.1028 std::unordered_set< _Value, _Hash, _Pred, _Alloc > Class Template Reference

Public Types

- typedef _Hashtable::key_type [key_type](#)
- typedef _Hashtable::value_type [value_type](#)
- typedef _Hashtable::hasher [hasher](#)
- typedef _Hashtable::key_equal [key_equal](#)
- typedef _Hashtable::allocator_type [allocator_type](#)
- typedef _Hashtable::pointer [pointer](#)
- typedef _Hashtable::const_pointer [const_pointer](#)
- typedef _Hashtable::reference [reference](#)
- typedef _Hashtable::const_reference [const_reference](#)
- typedef _Hashtable::iterator [iterator](#)
- typedef _Hashtable::const_iterator [const_iterator](#)
- typedef _Hashtable::local_iterator [local_iterator](#)
- typedef _Hashtable::const_local_iterator [const_local_iterator](#)
- typedef _Hashtable::size_type [size_type](#)
- typedef _Hashtable::difference_type [difference_type](#)

Public Member Functions

- `unordered_set` ()=default
- `unordered_set` (`size_type` __n, const `hasher` &__hf=`hasher`(), const `key_equal` &__eq|=`key_equal`(), const `allocator_type` &__a=`allocator_type`())
- `template<typename _InputIterator >`
`unordered_set` (_InputIterator __first, _InputIterator __last, `size_type` __n=0, const `hasher` &__hf=`hasher`(), const `key_equal` &__eq|=`key_equal`(), const `allocator_type` &__a=`allocator_type`())
- `unordered_set` (const `unordered_set` &)=default
- `unordered_set` (`unordered_set` &&)=default
- `unordered_set` (const `allocator_type` &__a)
- `unordered_set` (const `unordered_set` &__uset, const `allocator_type` &__a)
- `unordered_set` (`unordered_set` && __uset, const `allocator_type` &__a)
- `unordered_set` (`initializer_list`< `value_type` > __l, `size_type` __n=0, const `hasher` &__hf=`hasher`(), const `key_`←
`equal` &__eq|=`key_equal`(), const `allocator_type` &__a=`allocator_type`())
- `unordered_set` (`size_type` __n, const `allocator_type` &__a)
- `unordered_set` (`size_type` __n, const `hasher` &__hf, const `allocator_type` &__a)
- `template<typename _InputIterator >`
`unordered_set` (_InputIterator __first, _InputIterator __last, `size_type` __n, const `allocator_type` &__a)
- `template<typename _InputIterator >`
`unordered_set` (_InputIterator __first, _InputIterator __last, `size_type` __n, const `hasher` &__hf, const `allocator_`←
`_type` &__a)
- `unordered_set` (`initializer_list`< `value_type` > __l, `size_type` __n, const `allocator_type` &__a)
- `unordered_set` (`initializer_list`< `value_type` > __l, `size_type` __n, const `hasher` &__hf, const `allocator_type` &__a)
- `size_type bucket` (const `key_type` &__key) const
- `size_type bucket_count` () const noexcept
- `size_type bucket_size` (`size_type` __n) const
- `const_iterator cbegin` () const noexcept
- `const_iterator cend` () const noexcept
- `void clear` () noexcept
- `size_type count` (const `key_type` &__x) const
- `template<typename... _Args>`
`std::pair`< `iterator`, bool > `emplace` (_Args &&...__args)
- `template<typename... _Args>`
`iterator emplace_hint` (const `iterator` __pos, _Args &&...__args)
- `bool empty` () const noexcept
- `size_type erase` (const `key_type` &__x)
- `iterator erase` (const `iterator` __first, const `iterator` __last)
- `allocator_type get_allocator` () const noexcept
- `hasher hash_function` () const
- `template<typename _InputIterator >`
`void insert` (_InputIterator __first, _InputIterator __last)
- `void insert` (`initializer_list`< `value_type` > __l)
- `key_equal key_eq` () const
- `float load_factor` () const noexcept
- `size_type max_bucket_count` () const noexcept
- `float max_load_factor` () const noexcept
- `void max_load_factor` (float __z)
- `size_type max_size` () const noexcept
- `unordered_set & operator=` (const `unordered_set` &)=default
- `unordered_set & operator=` (`unordered_set` &&)=default

- `unordered_set` & `operator=` (`initializer_list`< `value_type` > `__l`)
- `void rehash` (`size_type` `__n`)
- `void reserve` (`size_type` `__n`)
- `size_type size` () `const` `noexcept`
- `void swap` (`unordered_set` & `__x`) `noexcept`(`noexcept`(`_M_h.swap`(`__x._M_h`)))
- `iterator begin` () `noexcept`
- `const_iterator begin` () `const` `noexcept`
- `iterator end` () `noexcept`
- `const_iterator end` () `const` `noexcept`
- `std::pair`< `iterator`, `bool` > `insert` (`const value_type` & `__x`)
- `std::pair`< `iterator`, `bool` > `insert` (`value_type` && `__x`)
- `iterator insert` (`const_iterator` `__hint`, `const value_type` & `__x`)
- `iterator insert` (`const_iterator` `__hint`, `value_type` && `__x`)
- `iterator erase` (`const_iterator` `__position`)
- `iterator erase` (`iterator` `__position`)
- `iterator find` (`const key_type` & `__x`)
- `const_iterator find` (`const key_type` & `__x`) `const`
- `std::pair`< `iterator`, `iterator` > `equal_range` (`const key_type` & `__x`)
- `std::pair`< `const_iterator`, `const_iterator` > `equal_range` (`const key_type` & `__x`) `const`
- `local_iterator begin` (`size_type` `__n`)
- `const_local_iterator begin` (`size_type` `__n`) `const`
- `const_local_iterator cbegin` (`size_type` `__n`) `const`
- `local_iterator end` (`size_type` `__n`)
- `const_local_iterator end` (`size_type` `__n`) `const`
- `const_local_iterator cend` (`size_type` `__n`) `const`

Friends

- `template`<`typename` `_Value1` , `typename` `_Hash1` , `typename` `_Pred1` , `typename` `_Alloc1` >
`bool operator==` (`const unordered_set`< `_Value1`, `_Hash1`, `_Pred1`, `_Alloc1` > &, `const unordered_set`< `_Value1`, `_Hash1`, `_Pred1`, `_Alloc1` > &)

5.1028.1 Detailed Description

```
template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>>
class std::unordered_set< _Value, _Hash, _Pred, _Alloc >
```

A standard container composed of unique keys (containing at most one of each key value) in which the elements' keys are the elements themselves.

Template Parameters

<code>_Value</code>	Type of key objects.
<code>_Hash</code>	Hashing function object type, defaults to <code>hash<_Value></code> .
<code>_Pred</code>	Predicate function object type, defaults to <code>equal_to<_Value></code> .
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_Key></code> .

Meets the requirements of a `container`, and `unordered associative container`

Base is `_Hashtable`, dispatched at compile time via template alias `__uset_hashtable`.

Definition at line 93 of file `unordered_set.h`.

5.1028.2 Member Typedef Documentation

5.1028.2.1 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> typedef _Hashtable::allocator_type std::unordered_set<_Value, _Hash, _Pred, _Alloc>::allocator_type`

Public typedefs.

Definition at line 106 of file `unordered_set.h`.

5.1028.2.2 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> typedef _Hashtable::const_iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc>::const_iterator`

Iterator-related typedefs.

Definition at line 116 of file `unordered_set.h`.

5.1028.2.3 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> typedef _Hashtable::const_local_iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc>::const_local_iterator`

Iterator-related typedefs.

Definition at line 118 of file `unordered_set.h`.

5.1028.2.4 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> typedef _Hashtable::const_pointer std::unordered_set<_Value, _Hash, _Pred, _Alloc>::const_pointer`

Iterator-related typedefs.

Definition at line 112 of file `unordered_set.h`.

```
5.1028.2.5 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =  
std::allocator<_Value>> typedef _Hashtable::const_reference std::unordered_set<_Value, _Hash, _Pred, _Alloc  
>::const_reference
```

Iterator-related typedefs.

Definition at line 114 of file `unordered_set.h`.

```
5.1028.2.6 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =  
std::allocator<_Value>> typedef _Hashtable::difference_type std::unordered_set<_Value, _Hash, _Pred, _Alloc  
>::difference_type
```

Iterator-related typedefs.

Definition at line 120 of file `unordered_set.h`.

```
5.1028.2.7 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =  
std::allocator<_Value>> typedef _Hashtable::hasher std::unordered_set<_Value, _Hash, _Pred, _Alloc  
>::hasher
```

Public typedefs.

Definition at line 104 of file `unordered_set.h`.

```
5.1028.2.8 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =  
std::allocator<_Value>> typedef _Hashtable::iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc  
>::iterator
```

Iterator-related typedefs.

Definition at line 115 of file `unordered_set.h`.

```
5.1028.2.9 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =  
std::allocator<_Value>> typedef _Hashtable::key_equal std::unordered_set<_Value, _Hash, _Pred, _Alloc  
>::key_equal
```

Public typedefs.

Definition at line 105 of file `unordered_set.h`.

```
5.1028.2.10 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =  
std::allocator<_Value>> typedef _Hashtable::key_type std::unordered_set<_Value, _Hash, _Pred, _Alloc  
>::key_type
```

Public typedefs.

Definition at line 102 of file `unordered_set.h`.

5.1028.2.11 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> typedef _Hashtable::local_iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc>::local_iterator`

Iterator-related typedefs.

Definition at line 117 of file `unordered_set.h`.

5.1028.2.12 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> typedef _Hashtable::pointer std::unordered_set<_Value, _Hash, _Pred, _Alloc>::pointer`

Iterator-related typedefs.

Definition at line 111 of file `unordered_set.h`.

5.1028.2.13 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> typedef _Hashtable::reference std::unordered_set<_Value, _Hash, _Pred, _Alloc>::reference`

Iterator-related typedefs.

Definition at line 113 of file `unordered_set.h`.

5.1028.2.14 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> typedef _Hashtable::size_type std::unordered_set<_Value, _Hash, _Pred, _Alloc>::size_type`

Iterator-related typedefs.

Definition at line 119 of file `unordered_set.h`.

5.1028.2.15 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> typedef _Hashtable::value_type std::unordered_set<_Value, _Hash, _Pred, _Alloc>::value_type`

Public typedefs.

Definition at line 103 of file `unordered_set.h`.

5.1028.3 Constructor & Destructor Documentation

5.1028.3.1 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> std::unordered_set<_Value, _Hash, _Pred, _Alloc>::unordered_set ()`
[default]

Default constructor.

Referenced by `std::unordered_set<_Value, _Hash, _Pred, _Alloc>::unordered_set()`.

5.1028.3.2 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> std::unordered_set<_Value, _Hash, _Pred, _Alloc>::unordered_set (size_type __n, const hasher & __hf = hasher(), const key_equal & __eq = key_equal(), const allocator_type & __a = allocator_type())` [inline], [explicit]

Default constructor creates no elements.

Parameters

<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eqf</code>	A key equality functor.
<code>__a</code>	An allocator object.

Definition at line 136 of file unordered_set.h.

5.1028.3.3 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> template<typename InputIterator> std::unordered_set<_Value, _Hash, _Pred, _Alloc>::unordered_set (InputIterator __first, InputIterator __last, size_type __n = 0, const hasher & __hf = hasher(), const key_equal & __eqf = key_equal(), const allocator_type & __a = allocator_type())`
`[inline]`

Builds an unordered_set from a range.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.
<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eqf</code>	A key equality functor.
<code>__a</code>	An allocator object.

Create an unordered_set consisting of copies of the elements from [__first,__last). This is linear in N (where N is distance(__first,__last)).

Definition at line 157 of file unordered_set.h.

References std::unordered_set< _Value, _Hash, _Pred, _Alloc >::unordered_set().

5.1028.3.4 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> std::unordered_set<_Value, _Hash, _Pred, _Alloc>::unordered_set (const unordered_set<_Value, _Hash, _Pred, _Alloc> &)` `[default]`

Copy constructor.

5.1028.3.5 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> std::unordered_set<_Value, _Hash, _Pred, _Alloc>::unordered_set (unordered_set<_Value, _Hash, _Pred, _Alloc> &&)` `[default]`

Move constructor.

5.1028.3.6 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> std::unordered_set<_Value, _Hash, _Pred, _Alloc>::unordered_set (const allocator_type & __a)` `[inline], [explicit]`

Creates an unordered_set with no elements.

Parameters

<code>__a</code>	An allocator object.
------------------	----------------------

Definition at line 176 of file `unordered_set.h`.

References `std::unordered_set<_Value, _Hash, _Pred, _Alloc>::unordered_set()`.

```
5.1028.3.7 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc
= std::allocator<_Value>> std::unordered_set<_Value, _Hash, _Pred, _Alloc>::unordered_set (
initializer_list<value_type> __l, size_type __n = 0, const hasher & __hf = hasher(), const key_equal &
__eqf = key_equal(), const allocator_type & __a = allocator_type()) [inline]
```

Builds an `unordered_set` from an `initializer_list`.

Parameters

<code>__l</code>	An <code>initializer_list</code> .
<code>__n</code>	Minimal initial number of buckets.
<code>__hf</code>	A hash functor.
<code>__eqf</code>	A key equality functor.
<code>__a</code>	An allocator object.

Create an `unordered_set` consisting of copies of the elements in the list. This is linear in N (where N is `__l.size()`).

Definition at line 211 of file `unordered_set.h`.

References `std::unordered_set<_Value, _Hash, _Pred, _Alloc>::operator=()`, and `std::unordered_set<_Value, __hf, _Pred, _Alloc>::unordered_set()`.

5.1028.4 Member Function Documentation

```
5.1028.4.1 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc>::begin() [inline],
[noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the `unordered_set`.

Definition at line 311 of file `unordered_set.h`.

```
5.1028.4.2 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> const_iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc>::begin() const
[inline], [noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the `unordered_set`.

Definition at line 315 of file `unordered_set.h`.

5.1028.4.3 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> local_iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc>::begin (size_type __n) [inline]`

Returns a read-only (constant) iterator pointing to the first bucket element.

Parameters

$_n$	The bucket index.
-------	-------------------

Returns

A read-only local iterator.

Definition at line 662 of file unordered_set.h.

```
5.1028.4.4  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> const_local_iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc>::begin (
size_type __n ) const  [inline]
```

Returns a read-only (constant) iterator pointing to the first bucket element.

Parameters

$_n$	The bucket index.
-------	-------------------

Returns

A read-only local iterator.

Definition at line 666 of file unordered_set.h.

```
5.1028.4.5  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> size_type std::unordered_set<_Value, _Hash, _Pred, _Alloc>::bucket_count ( )
const  [inline],[noexcept]
```

Returns the number of buckets of the unordered_set.

Definition at line 628 of file unordered_set.h.

```
5.1028.4.6  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> const_iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc>::cbegin ( ) const
[inline],[noexcept]
```

Returns a read-only (constant) iterator that points to the first element in the unordered_set.

Definition at line 338 of file unordered_set.h.

```
5.1028.4.7  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> const_local_iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc>::cbegin (
size_type __n ) const  [inline]
```

Returns a read-only (constant) iterator pointing to the first bucket element.

Parameters

$_n$	The bucket index.
-------	-------------------

Returns

A read-only local iterator.

Definition at line 670 of file unordered_set.h.

```
5.1028.4.8  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
            std::allocator<_Value>> const_iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc>::end ( ) const
            [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the unordered_set.

Definition at line 346 of file unordered_set.h.

```
5.1028.4.9  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
            std::allocator<_Value>> const_local_iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc>::end (
            size_type __n ) const [inline]
```

Returns a read-only (constant) iterator pointing to one past the last bucket elements.

Parameters

$_n$	The bucket index.
-------	-------------------

Returns

A read-only local iterator.

Definition at line 690 of file unordered_set.h.

```
5.1028.4.10 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
            std::allocator<_Value>> void std::unordered_set<_Value, _Hash, _Pred, _Alloc>::clear ( ) [inline],
            [noexcept]
```

Erases all elements in an unordered_set. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 539 of file unordered_set.h.

```
5.1028.4.11 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
            std::allocator<_Value>> size_type std::unordered_set<_Value, _Hash, _Pred, _Alloc>::count ( const
            key_type & __x ) const [inline]
```

Finds the number of elements.

Parameters

<code>_↔</code>	Element to located.
<code>_X</code>	

Returns

Number of elements with specified key.

This function only makes sense for `unordered_multisets`; for `unordered_set` the result will either be 0 (not present) or 1 (present).

Definition at line 603 of file `unordered_set.h`.

```
5.1028.4.12  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> template<typename... _Args> std::pair<iterator, bool> std::unordered_set<
              _Value, _Hash, _Pred, _Alloc >::emplace ( _Args &&... __args ) [inline]
```

Attempts to build and insert an element into the `unordered_set`.

Parameters

<code>__args</code>	Arguments used to generate an element.
---------------------	--

Returns

A pair, of which the first element is an iterator that points to the possibly inserted element, and the second is a `bool` that is true if the element was actually inserted.

This function attempts to build and insert an element into the `unordered_set`. An `unordered_set` relies on unique keys and thus an element is only inserted if it is not already present in the `unordered_set`.

Insertion requires amortized constant time.

Definition at line 368 of file `unordered_set.h`.

```
5.1028.4.13  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> template<typename... _Args> iterator std::unordered_set< _Value, _Hash, _Pred,
              _Alloc >::emplace_hint ( const_iterator __pos, _Args &&... __args ) [inline]
```

Attempts to insert an element into the `unordered_set`.

Parameters

<code>__pos</code>	An iterator that serves as a hint as to where the element should be inserted.
<code>__args</code>	Arguments used to generate the element to be inserted.

Returns

An iterator that points to the element with key equivalent to the one generated from `__args` (may or may not be the element itself).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `emplace()` does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.↵html#containers.associative.insert_hints

Insertion requires amortized constant time.

Definition at line 394 of file `unordered_set.h`.

```
5.1028.4.14  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
             std::allocator<_Value>> bool std::unordered_set<_Value, _Hash, _Pred, _Alloc>::empty ( ) const
             [inline], [noexcept]
```

Returns true if the `unordered_set` is empty.

Definition at line 290 of file `unordered_set.h`.

```
5.1028.4.15  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
             std::allocator<_Value>> iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc>::end ( ) [inline],
             [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the `unordered_set`.

Definition at line 325 of file `unordered_set.h`.

```
5.1028.4.16  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
             std::allocator<_Value>> const_iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc>::end ( ) const
             [inline], [noexcept]
```

Returns a read-only (constant) iterator that points one past the last element in the `unordered_set`.

Definition at line 329 of file `unordered_set.h`.

```
5.1028.4.17  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
             std::allocator<_Value>> local_iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc>::end ( size_type
             __n ) [inline]
```

Returns a read-only (constant) iterator pointing to one past the last bucket elements.

Parameters

<code>__↵</code>	The bucket index.
<code>__n</code>	

Returns

A read-only local iterator.

Definition at line 682 of file unordered_set.h.

```
5.1028.4.18  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> const_local_iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc >::end (
              size_type __n ) const    [inline]
```

Returns a read-only (constant) iterator pointing to one past the last bucket elements.

Parameters

$_n$	The bucket index.
-------	-------------------

Returns

A read-only local iterator.

Definition at line 686 of file unordered_set.h.

```
5.1028.4.19  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> std::pair<iterator, iterator> std::unordered_set<_Value, _Hash, _Pred, _Alloc
              >::equal_range ( const key_type & __x )    [inline]
```

Finds a subsequence matching given key.

Parameters

$_x$	Key to be located.
-------	--------------------

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function probably only makes sense for multisets.

Definition at line 616 of file unordered_set.h.

```
5.1028.4.20  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> std::pair<const_iterator, const_iterator> std::unordered_set<_Value, _Hash,
              _Pred, _Alloc >::equal_range ( const key_type & __x ) const    [inline]
```

Finds a subsequence matching given key.

Parameters

<code>_↔</code>	Key to be located.
<code>_X</code>	

Returns

Pair of iterators that possibly points to the subsequence matching given key.

This function probably only makes sense for multisets.

Definition at line 620 of file unordered_set.h.

```
5.1028.4.21  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =  
std::allocator<_Value>> iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc>::erase ( const_iterator  
__position ) [inline]
```

Erases an element from an unordered_set.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following `__position` prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from an unordered_set. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 489 of file unordered_set.h.

```
5.1028.4.22  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =  
std::allocator<_Value>> iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc>::erase ( iterator  
__position ) [inline]
```

Erases an element from an unordered_set.

Parameters

<code>__position</code>	An iterator pointing to the element to be erased.
-------------------------	---

Returns

An iterator pointing to the element immediately following `__position` prior to the element being erased. If no such element exists, `end()` is returned.

This function erases an element, pointed to by the given iterator, from an `unordered_set`. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 494 of file `unordered_set.h`.

```
5.1028.4.23  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> size_type std::unordered_set< _Value, _Hash, _Pred, _Alloc >::erase ( const
              key_type & __x ) [inline]
```

Erases elements according to the provided key.

Parameters

<code>__x</code>	Key of element to be erased.
------------------	------------------------------

Returns

The number of elements erased.

This function erases all the elements located by the given key from an `unordered_set`. For an `unordered_set` the result of this function can only be 0 (not present) or 1 (present). Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 511 of file `unordered_set.h`.

```
5.1028.4.24  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> iterator std::unordered_set< _Value, _Hash, _Pred, _Alloc >::erase ( const_iterator
              __first, const_iterator __last ) [inline]
```

Erases a `[__first, __last)` range of elements from an `unordered_set`.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be erased.
<code>__last</code>	Iterator pointing to the end of the range to be erased.

Returns

The iterator `__last`.

This function erases a sequence of elements from an unordered_set. Note that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 529 of file unordered_set.h.

5.1028.4.25 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc>::find (const key_type &__x) [inline]`

Tries to locate an element in an unordered_set.

Parameters

<code>__x</code>	Element to be located.
------------------	------------------------

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (end()) iterator.

Definition at line 585 of file unordered_set.h.

5.1028.4.26 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> const_iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc>::find (const key_type &__x) const [inline]`

Tries to locate an element in an unordered_set.

Parameters

<code>__x</code>	Element to be located.
------------------	------------------------

Returns

Iterator pointing to sought-after element, or end() if not found.

This function takes a key and tries to locate the element with which the key matches. If successful the function returns an iterator pointing to the sought after element. If unsuccessful it returns the past-the-end (end()) iterator.

Definition at line 589 of file unordered_set.h.


```
5.1028.4.27 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> allocator_type std::unordered_set<_Value, _Hash, _Pred, _Alloc>::get_allocator (
) const [inline], [noexcept]
```

Returns the allocator object with which the unordered_set was constructed.

Definition at line 283 of file unordered_set.h.

```
5.1028.4.28 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> hasher std::unordered_set<_Value, _Hash, _Pred, _Alloc>::hash_function ( ) const
[inline]
```



Returns the hash functor object with which the unordered_set was constructed.

Definition at line 561 of file unordered_set.h.

```
5.1028.4.29 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> std::pair<iterator, bool> std::unordered_set<_Value, _Hash, _Pred, _Alloc>
::insert( const value_type &__x ) [inline]
```

Attempts to insert an element into the unordered_set.

Parameters

	Element to be inserted.
	

Returns

A pair, of which the first element is an iterator that points to the possibly inserted element, and the second is a bool that is true if the element was actually inserted.

This function attempts to insert an element into the unordered_set. An unordered_set relies on unique keys and thus an element is only inserted if it is not already present in the unordered_set.



Insertion requires amortized constant time.

Definition at line 412 of file unordered_set.h.

```
5.1028.4.30 template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
std::allocator<_Value>> std::pair<iterator, bool> std::unordered_set<_Value, _Hash, _Pred, _Alloc>
::insert( value_type &&__x ) [inline]
```

Attempts to insert an element into the unordered_set.

Parameters

	Element to be inserted.
	

Returns

A pair, of which the first element is an iterator that points to the possibly inserted element, and the second is a bool that is true if the element was actually inserted.

This function attempts to insert an element into the unordered_set. An unordered_set relies on unique keys and thus an element is only inserted if it is not already present in the unordered_set.

Insertion requires amortized constant time.

Definition at line 416 of file unordered_set.h.

```
5.1028.4.31  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc>::insert ( const_iterator
              __hint, const value_type & __x ) [inline]
```

Attempts to insert an element into the unordered_set.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the element should be inserted.
<code>__x</code>	Element to be inserted.

Returns

An iterator that points to the element with key of `__x` (may or may not be the element passed in).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `insert()` does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.html#containers.associative.insert_hints

Insertion requires amortized constant.

Definition at line 441 of file unordered_set.h.

```
5.1028.4.32  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> iterator std::unordered_set<_Value, _Hash, _Pred, _Alloc>::insert ( const_iterator
              __hint, value_type && __x ) [inline]
```

Attempts to insert an element into the unordered_set.

Parameters

<code>__hint</code>	An iterator that serves as a hint as to where the element should be inserted.
<code>__x</code>	Element to be inserted.

Returns

An iterator that points to the element with key of `__x` (may or may not be the element passed in).

This function is not concerned about whether the insertion took place, and thus does not return a boolean like the single-argument `insert()` does. Note that the first parameter is only a hint and can potentially improve the performance of the insertion process. A bad hint would cause no gains in efficiency.

For more on *hinting*, see: https://gcc.gnu.org/onlinedocs/libstdc++/manual/associative.↵html#containers.associative.insert_hints

Insertion requires amortized constant.

Definition at line 445 of file `unordered_set.h`.

```
5.1028.4.33  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> template<typename _InputIterator > void std::unordered_set<_Value, _Hash, _Pred,
              _Alloc >::insert ( _InputIterator __first, _InputIterator __last ) [inline]
```

A template function that attempts to insert a range of elements.

Parameters

<code>__first</code>	Iterator pointing to the start of the range to be inserted.
<code>__last</code>	Iterator pointing to the end of the range.

Complexity similar to that of the range constructor.

Definition at line 460 of file `unordered_set.h`.

```
5.1028.4.34  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> void std::unordered_set<_Value, _Hash, _Pred, _Alloc >::insert ( initializer_list<
              value_type > __l ) [inline]
```

Attempts to insert a list of elements into the `unordered_set`.

Parameters

↵ _↵ ↵ _↵ /	A <code>std::initializer_list<value_type></code> of elements to be inserted.
-------------------------	--

Complexity similar to that of the range constructor.

Definition at line 471 of file `unordered_set.h`.

5.1028.4.35 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> key_equal std::unordered_set<_Value, _Hash, _Pred, _Alloc>::key_eq () const [inline]`

Returns the key comparison object with which the unordered_set was constructed.

Definition at line 567 of file unordered_set.h.

5.1028.4.36 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> float std::unordered_set<_Value, _Hash, _Pred, _Alloc>::load_factor () const [inline], [noexcept]`

Returns the average number of elements per bucket.

Definition at line 698 of file unordered_set.h.

5.1028.4.37 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> size_type std::unordered_set<_Value, _Hash, _Pred, _Alloc>::max_bucket_count () const [inline], [noexcept]`

Returns the maximum number of buckets of the unordered_set.

Definition at line 633 of file unordered_set.h.

5.1028.4.38 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> float std::unordered_set<_Value, _Hash, _Pred, _Alloc>::max_load_factor () const [inline], [noexcept]`

Returns a positive number that the unordered_set tries to keep the load factor less than or equal to.

Definition at line 704 of file unordered_set.h.

5.1028.4.39 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> void std::unordered_set<_Value, _Hash, _Pred, _Alloc>::max_load_factor (float __z) [inline]`

Change the unordered_set maximum load factor.

Parameters

<code>__z</code>	The new maximum load factor.
------------------	------------------------------

Definition at line 712 of file unordered_set.h.

5.1028.4.40 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> size_type std::unordered_set<_Value, _Hash, _Pred, _Alloc>::max_size () const [inline], [noexcept]`

Returns the maximum size of the unordered_set.

Definition at line 300 of file unordered_set.h.

```
5.1028.4.41  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> unordered_set& std::unordered_set<_Value, _Hash, _Pred, _Alloc>::operator= (
              const unordered_set<_Value, _Hash, _Pred, _Alloc> & ) [default]
```

Copy assignment operator.

Referenced by std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::unordered_multiset(), and std::unordered_↵
_set<_Value, _Hash, _Pred, _Alloc>::unordered_set().

```
5.1028.4.42  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> unordered_set& std::unordered_set<_Value, _Hash, _Pred, _Alloc>::operator= (
              unordered_set<_Value, _Hash, _Pred, _Alloc> && ) [default]
```

Move assignment operator.

```
5.1028.4.43  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> unordered_set& std::unordered_set<_Value, _Hash, _Pred, _Alloc>::operator= (
              initializer_list<value_type> __l ) [inline]
```

Unordered_set list assignment operator.

Parameters

↵	An initializer_list.
__l	
↵	
__l	
l	

This function fills an unordered_set with copies of the elements in the initializer list __l.

Note that the assignment completely changes the unordered_set and that the resulting unordered_set's size is the same as the number of elements assigned. Old data may be lost.

Definition at line 274 of file unordered_set.h.

```
5.1028.4.44  template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc =
              std::allocator<_Value>> void std::unordered_set<_Value, _Hash, _Pred, _Alloc>::rehash ( size_type __n )
              [inline]
```

May rehash the unordered_set.

Parameters

↵	The new number of buckets.
__n	

Rehash will occur only if the new number of buckets respect the unordered_set maximum load factor.

Definition at line 723 of file unordered_set.h.

5.1028.4.45 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> void std::unordered_set<_Value, _Hash, _Pred, _Alloc>::reserve(size_type __n)`
`[inline]`

Prepare the unordered_set for a specified number of elements.

Parameters

<code>__n</code>	Number of elements required.
------------------	------------------------------

Same as `rehash(ceil(n / max_load_factor()))`.

Definition at line 734 of file unordered_set.h.

5.1028.4.46 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> size_type std::unordered_set<_Value, _Hash, _Pred, _Alloc>::size() const`
`[inline], [noexcept]`

Returns the size of the unordered_set.

Definition at line 295 of file unordered_set.h.

5.1028.4.47 `template<class _Value, class _Hash = hash<_Value>, class _Pred = std::equal_to<_Value>, class _Alloc = std::allocator<_Value>> void std::unordered_set<_Value, _Hash, _Pred, _Alloc>::swap(unordered_set<_Value, _Hash, _Pred, _Alloc> &__x)` `[inline], [noexcept]`

Swaps data with another unordered_set.

Parameters

<code>__x</code>	An unordered_set of the same element and allocator types.
------------------	---

This exchanges the elements between two sets in constant time. Note that the global `std::swap()` function is specialized such that `std::swap(s1,s2)` will feed to this function.

Definition at line 552 of file unordered_set.h.

Referenced by `std::unordered_multiset<_Value, _Hash, _Pred, _Alloc>::reserve()`.

The documentation for this class was generated from the following file:

- [unordered_set.h](#)

5.1029 `std::uses_allocator<_Tp, _Alloc>` Struct Template Reference

Inherits `type<_Tp, _Alloc>`.

5.1029.1 Detailed Description

```
template<typename _Tp, typename _Alloc>
struct std::uses_allocator<_Tp, _Alloc>
```

Declare `uses_allocator` so it can be specialized in `<queue>` etc.

[allocator.uses.trait]

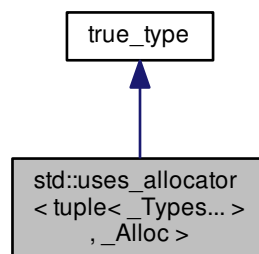
Definition at line 71 of file `memoryfwd.h`.

The documentation for this struct was generated from the following file:

- [memoryfwd.h](#)

5.1030 `std::uses_allocator<tuple<_Types...>, _Alloc>` Struct Template Reference

Inheritance diagram for `std::uses_allocator<tuple<_Types...>, _Alloc>`:



Public Types

- typedef `integral_constant<_Tp, __v>` **type**
- typedef `_Tp` **value_type**

Public Member Functions

- constexpr **operator value_type** () const
- constexpr value_type **operator()** () const

Static Public Attributes

- static constexpr **_Tp value**

5.1030.1 Detailed Description

```
template<typename... _Types, typename _Alloc>
struct std::uses_allocator< tuple< _Types... >, _Alloc >
```

Partial specialization for tuples.

Definition at line 1564 of file tuple.

The documentation for this struct was generated from the following file:

- [tuple](#)

5.1031 `std::valarray<_Tp>` Class Template Reference

Public Types

- typedef **_Tp value_type**

Public Member Functions

- [valarray](#) ()
- [valarray](#) (size_t)
- [valarray](#) (const _Tp &, size_t)
- [valarray](#) (const _Tp *__restrict__, size_t)
- [valarray](#) (const [valarray](#) &)
- [valarray](#) ([valarray](#) &&) noexcept
- [valarray](#) (const [slice_array](#)<_Tp> &)
- [valarray](#) (const [gslice_array](#)<_Tp> &)
- [valarray](#) (const [mask_array](#)<_Tp> &)
- [valarray](#) (const [indirect_array](#)<_Tp> &)
- [valarray](#) ([initializer_list](#)<_Tp>)
- template<class _Dom>
 valarray (const _Expr<_Dom, _Tp> &__e)
- template<typename _Tp>
 valarray (const _Tp *__restrict__ __p, size_t __n)
- _Expr<_ValFunClos<_ValArray, _Tp>, _Tp> [apply](#) (_Tp func(_Tp)) const

- `_Expr< _RefFunClos< _ValArray, _Tp >, _Tp > apply` (`_Tp func(const _Tp &)`) `const`
- `valarray< _Tp > cshift` (`int __n`) `const`
- `_Tp max` () `const`
- `_Tp min` () `const`
- `_UnaryOp< __logical_not >::_Rt operator!` () `const`
- `valarray< _Tp > & operator%=>` (`const _Tp &`)
- `valarray< _Tp > & operator%=>` (`const valarray< _Tp > &`)
- `template<class _Dom >`
`valarray< _Tp > & operator%=>` (`const _Expr< _Dom, _Tp > &`)
- `valarray< _Tp > & operator&=>` (`const _Tp &`)
- `valarray< _Tp > & operator&=>` (`const valarray< _Tp > &`)
- `template<class _Dom >`
`valarray< _Tp > & operator&=>` (`const _Expr< _Dom, _Tp > &`)
- `valarray< _Tp > & operator*=>` (`const _Tp &`)
- `valarray< _Tp > & operator*=>` (`const valarray< _Tp > &`)
- `template<class _Dom >`
`valarray< _Tp > & operator*=>` (`const _Expr< _Dom, _Tp > &`)
- `_UnaryOp< __unary_plus >::_Rt operator+` () `const`
- `valarray< _Tp > & operator+=>` (`const _Tp &`)
- `valarray< _Tp > & operator+=>` (`const valarray< _Tp > &`)
- `template<class _Dom >`
`valarray< _Tp > & operator+=>` (`const _Expr< _Dom, _Tp > &`)
- `_UnaryOp< __negate >::_Rt operator-` () `const`
- `valarray< _Tp > & operator-=>` (`const _Tp &`)
- `valarray< _Tp > & operator-=>` (`const valarray< _Tp > &`)
- `template<class _Dom >`
`valarray< _Tp > & operator-=>` (`const _Expr< _Dom, _Tp > &`)
- `valarray< _Tp > & operator/=>` (`const _Tp &`)
- `valarray< _Tp > & operator/=>` (`const valarray< _Tp > &`)
- `template<class _Dom >`
`valarray< _Tp > & operator/=>` (`const _Expr< _Dom, _Tp > &`)
- `valarray< _Tp > & operator<=>` (`const _Tp &`)
- `valarray< _Tp > & operator<=>` (`const valarray< _Tp > &`)
- `template<class _Dom >`
`valarray< _Tp > & operator<=>` (`const _Expr< _Dom, _Tp > &`)
- `valarray< _Tp > & operator=>` (`const valarray< _Tp > &__v`)
- `valarray< _Tp > & operator=>` (`valarray< _Tp > &&__v`) `noexcept`
- `valarray< _Tp > & operator=>` (`const _Tp &__t`)
- `valarray< _Tp > & operator=>` (`const slice_array< _Tp > &__sa`)
- `valarray< _Tp > & operator=>` (`const gslice_array< _Tp > &__ga`)
- `valarray< _Tp > & operator=>` (`const mask_array< _Tp > &__ma`)
- `valarray< _Tp > & operator=>` (`const indirect_array< _Tp > &__ia`)
- `valarray & operator=>` (`initializer_list< _Tp > __l`)
- `template<class _Dom >`
`valarray< _Tp > & operator=>` (`const _Expr< _Dom, _Tp > &`)
- `valarray< _Tp > & operator>=>` (`const _Tp &`)
- `valarray< _Tp > & operator>=>` (`const valarray< _Tp > &`)
- `template<class _Dom >`
`valarray< _Tp > & operator>=>` (`const _Expr< _Dom, _Tp > &`)
- `_Tp & operator[]` (`size_t __i`)
- `const _Tp & operator[]` (`size_t`) `const`

- `_Expr<_SClos<_ValArray, _Tp>, _Tp> operator[] (slice __s) const`
- `slice_array<_Tp> operator[] (slice __s)`
- `_Expr<_GClos<_ValArray, _Tp>, _Tp> operator[] (const gslice &__s) const`
- `gslice_array<_Tp> operator[] (const gslice &__s)`
- `valarray<_Tp> operator[] (const valarray<bool> &__m) const`
- `mask_array<_Tp> operator[] (const valarray<bool> &__m)`
- `_Expr<_IClos<_ValArray, _Tp>, _Tp> operator[] (const valarray<size_t> &__i) const`
- `indirect_array<_Tp> operator[] (const valarray<size_t> &__i)`
- `valarray<_Tp> & operator^= (const _Tp &)`
- `valarray<_Tp> & operator^= (const valarray<_Tp> &)`
- `template<class _Dom>`
`valarray<_Tp> & operator^= (const _Expr<_Dom, _Tp> &)`
- `valarray<_Tp> & operator|= (const _Tp &)`
- `valarray<_Tp> & operator|= (const valarray<_Tp> &)`
- `template<class _Dom>`
`valarray<_Tp> & operator|= (const _Expr<_Dom, _Tp> &)`
- `_UnaryOp<__bitwise_not>::Rt operator~ () const`
- `void resize (size_t __size, _Tp __c=_Tp())`
- `valarray<_Tp> shift (int __n) const`
- `size_t size () const`
- `_Tp sum () const`
- `void swap (valarray<_Tp> &__v) noexcept`

Friends

- `class _Array<_Tp>`

5.1031.1 Detailed Description

```
template<class _Tp>
class std::valarray<_Tp>
```

Smart array designed to support numeric processing.

A valarray is an array that provides constraints intended to allow for effective optimization of numeric array processing by reducing the aliasing that can result from pointer representations. It represents a one-dimensional array from which different multidimensional subsets can be accessed and modified.

Template Parameters

<code>_Tp</code>	Type of object in the array.
------------------	------------------------------

Definition at line 78 of file valarray.

5.1031.2 Constructor & Destructor Documentation

5.1031.2.1 `template<class _Tp> std::valarray<_Tp>::valarray (const _Tp * __restrict__, size_t)`

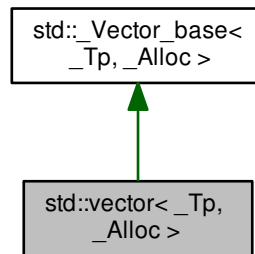
Construct an array initialized to the first n elements of t .

The documentation for this class was generated from the following file:

- [valarray](#)

5.1032 `std::vector<_Tp, _Alloc>` Class Template Reference

Inheritance diagram for `std::vector<_Tp, _Alloc>`:



Public Types

- typedef `_Alloc` **allocator_type**
- typedef `__gnu_cxx::__normal_iterator< const_pointer, vector >` **const_iterator**
- typedef `_Alloc_traits::const_pointer` **const_pointer**
- typedef `_Alloc_traits::const_reference` **const_reference**
- typedef `std::reverse_iterator< const_iterator >` **const_reverse_iterator**
- typedef `ptrdiff_t` **difference_type**
- typedef `__gnu_cxx::__normal_iterator< pointer, vector >` **iterator**
- typedef `_Base::pointer` **pointer**
- typedef `_Alloc_traits::reference` **reference**
- typedef `std::reverse_iterator< iterator >` **reverse_iterator**
- typedef `size_t` **size_type**
- typedef `_Tp` **value_type**

Public Member Functions

- [vector](#) () noexcept(is_nothrow_default_constructible<_Alloc>::value)
- [vector](#) (const allocator_type &__a) noexcept
- [vector](#) (size_type __n, const allocator_type &__a=allocator_type())
- [vector](#) (size_type __n, const value_type &__value, const allocator_type &__a=allocator_type())
- [vector](#) (const [vector](#) &__x)
- [vector](#) ([vector](#) &&__x) noexcept
- [vector](#) (const [vector](#) &__x, const allocator_type &__a)
- [vector](#) ([vector](#) &&__rv, const allocator_type &__m) noexcept(_Alloc_traits::S_always_equal())
- [vector](#) (initializer_list<value_type> __l, const allocator_type &__a=allocator_type())
- template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>>
[vector](#) (_InputIterator __first, _InputIterator __last, const allocator_type &__a=allocator_type())
- ~[vector](#) () noexcept
- void [assign](#) (size_type __n, const value_type &__val)
- template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>>
void [assign](#) (_InputIterator __first, _InputIterator __last)
- void [assign](#) (initializer_list<value_type> __l)
- reference [at](#) (size_type __n)
- const_reference [at](#) (size_type __n) const
- reference [back](#) () noexcept
- const_reference [back](#) () const noexcept
- iterator [begin](#) () noexcept
- const_iterator [begin](#) () const noexcept
- size_type [capacity](#) () const noexcept
- const_iterator [cbegin](#) () const noexcept
- const_iterator [cend](#) () const noexcept
- void [clear](#) () noexcept
- const_reverse_iterator [crbegin](#) () const noexcept
- const_reverse_iterator [crend](#) () const noexcept
- _Tp * [data](#) () noexcept
- const _Tp * [data](#) () const noexcept
- template<typename... _Args>
iterator [emplace](#) (const_iterator __position, _Args &&... __args)
- template<typename... _Args>
void [emplace_back](#) (_Args &&... __args)
- bool [empty](#) () const noexcept
- iterator [end](#) () noexcept
- const_iterator [end](#) () const noexcept
- iterator [erase](#) (const_iterator __position)
- iterator [erase](#) (const_iterator __first, const_iterator __last)
- reference [front](#) () noexcept
- const_reference [front](#) () const noexcept
- iterator [insert](#) (const_iterator __position, const value_type &__x)
- iterator [insert](#) (const_iterator __position, value_type &&__x)
- iterator [insert](#) (const_iterator __position, initializer_list<value_type> __l)
- iterator [insert](#) (const_iterator __position, size_type __n, const value_type &__x)
- template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>>
iterator [insert](#) (const_iterator __position, _InputIterator __first, _InputIterator __last)
- size_type [max_size](#) () const noexcept
- [vector](#) & [operator=](#) (const [vector](#) &__x)

- `vector & operator= (vector && __x) noexcept(_Alloc_traits::_S_nothrow_move())`
- `vector & operator= (initializer_list< value_type > __l)`
- `reference operator[] (size_type __n) noexcept`
- `const_reference operator[] (size_type __n) const noexcept`
- `void pop_back () noexcept`
- `void push_back (const value_type & __x)`
- `void push_back (value_type && __x)`
- `reverse_iterator rbegin () noexcept`
- `const_reverse_iterator rbegin () const noexcept`
- `reverse_iterator rend () noexcept`
- `const_reverse_iterator rend () const noexcept`
- `void reserve (size_type __n)`
- `void resize (size_type __new_size)`
- `void resize (size_type __new_size, const value_type & __x)`
- `void shrink_to_fit ()`
- `size_type size () const noexcept`
- `void swap (vector & __x) noexcept`

Protected Member Functions

- `pointer _M_allocate (size_t __n)`
- `template<typename _ForwardIterator >`
`pointer _M_allocate_and_copy (size_type __n, _ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _InputIterator >`
`void _M_assign_aux (_InputIterator __first, _InputIterator __last, std::input_iterator_tag)`
- `template<typename _ForwardIterator >`
`void _M_assign_aux (_ForwardIterator __first, _ForwardIterator __last, std::forward_iterator_tag)`
- `template<typename _Integer >`
`void _M_assign_dispatch (_Integer __n, _Integer __val, __true_type)`
- `template<typename _InputIterator >`
`void _M_assign_dispatch (_InputIterator __first, _InputIterator __last, __false_type)`
- `size_type _M_check_len (size_type __n, const char *__s) const`
- `void _M_deallocate (pointer __p, size_t __n)`
- `void _M_default_append (size_type __n)`
- `void _M_default_initialize (size_type __n)`
- `template<typename... _Args>`
`void _M_emplace_back_aux (_Args &&... __args)`
- `iterator _M_erase (iterator __position)`
- `iterator _M_erase (iterator __first, iterator __last)`
- `void _M_erase_at_end (pointer __pos) noexcept`
- `void _M_fill_assign (size_type __n, const value_type & __val)`
- `void _M_fill_initialize (size_type __n, const value_type & __value)`
- `void _M_fill_insert (iterator __pos, size_type __n, const value_type & __x)`
- `_Tp_alloc_type & _M_get_Tp_allocator () noexcept`
- `const _Tp_alloc_type & _M_get_Tp_allocator () const noexcept`
- `template<typename _Integer >`
`void _M_initialize_dispatch (_Integer __n, _Integer __value, __true_type)`
- `template<typename _InputIterator >`
`void _M_initialize_dispatch (_InputIterator __first, _InputIterator __last, __false_type)`

- `template<typename... _Args>`
`void _M_insert_aux (iterator __position, _Args &&... __args)`
- `template<typename _Integer >`
`void _M_insert_dispatch (iterator __pos, _Integer __n, _Integer __val, __true_type)`
- `template<typename _InputIterator >`
`void _M_insert_dispatch (iterator __pos, _InputIterator __first, _InputIterator __last, __false_type)`
- `void _M_range_check (size_type __n) const`
- `template<typename _InputIterator >`
`void _M_range_initialize (_InputIterator __first, _InputIterator __last, std::input_iterator_tag)`
- `template<typename _ForwardIterator >`
`void _M_range_initialize (_ForwardIterator __first, _ForwardIterator __last, std::forward_iterator_tag)`
- `template<typename _InputIterator >`
`void _M_range_insert (iterator __pos, _InputIterator __first, _InputIterator __last, std::input_iterator_tag)`
- `template<typename _ForwardIterator >`
`void _M_range_insert (iterator __pos, _ForwardIterator __first, _ForwardIterator __last, std::forward_iterator_tag)`
- `bool _M_shrink_to_fit ()`
- `allocator_type get_allocator () const noexcept`

Protected Attributes

- `_Vector_impl _M_impl`

5.1032.1 Detailed Description

```
template<typename _Tp, typename _Alloc = std::allocator<_Tp>>
class std::vector<_Tp, _Alloc >
```

A standard container which offers fixed time access to individual elements in any order.

Template Parameters

<code>_Tp</code>	Type of element.
<code>_Alloc</code>	Allocator type, defaults to <code>allocator<_Tp></code> .

Meets the requirements of a [container](#), a [reversible container](#), and a [sequence](#), including the [optional sequence requirements](#) with the exception of `push_front` and `pop_front`.

In some terminology a vector can be described as a dynamic C-style array, it offers fast and efficient access to individual elements in any order and saves the user from worrying about memory and size allocation. Subscripting (`[]`) access is also provided as with C-style arrays.

Definition at line 214 of file `stl_vector.h`.

5.1032.2 Constructor & Destructor Documentation

5.1032.2.1 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::vector<_Tp, _Alloc>::vector ()`
`[inline], [noexcept]`

Creates a vector with no elements.

Definition at line 255 of file `stl_vector.h`.

5.1032.2.2 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::vector<_Tp, _Alloc>::vector (const`
`allocator_type & __a) [inline], [explicit], [noexcept]`

Creates a vector with no elements.

Parameters

<code>__a</code>	An allocator object.
------------------	----------------------

Definition at line 266 of file `stl_vector.h`.

5.1032.2.3 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::vector<_Tp, _Alloc>::vector (size_type`
`__n, const allocator_type & __a = allocator_type()) [inline], [explicit]`

Creates a vector with default constructed elements.

Parameters

<code>__n</code>	The number of elements to initially create.
<code>__a</code>	An allocator.

This constructor fills the vector with `__n` default constructed elements.

Definition at line 279 of file `stl_vector.h`.

5.1032.2.4 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::vector<_Tp, _Alloc>::vector (size_type`
`__n, const value_type & __value, const allocator_type & __a = allocator_type()) [inline]`

Creates a vector with copies of an exemplar element.

Parameters

<code>__n</code>	The number of elements to initially create.
<code>__value</code>	An element to copy.
<code>__a</code>	An allocator.

This constructor fills the vector with `__n` copies of `__value`.

Definition at line 291 of file stl_vector.h.

5.1032.2.5 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::vector<_Tp, _Alloc>::vector (const vector<_Tp, _Alloc> & __x) [inline]`

Vector copy constructor.

Parameters

<code>__x</code>	A vector of identical element and allocator types.
------------------	--

The newly-created vector uses a copy of the allocation object used by `__x`. All the elements of `__x` are copied, but any extra memory in `__x` (for fast expansion) will not be copied.

Definition at line 320 of file stl_vector.h.

5.1032.2.6 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::vector<_Tp, _Alloc>::vector (vector<_Tp, _Alloc> && __x) [inline], [noexcept]`

Vector move constructor.

Parameters

<code>__x</code>	A vector of identical element and allocator types.
------------------	--

The newly-created vector contains the exact contents of `__x`. The contents of `__x` are a valid, but unspecified vector.

Definition at line 337 of file stl_vector.h.

5.1032.2.7 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::vector<_Tp, _Alloc>::vector (const vector<_Tp, _Alloc> & __x, const allocator_type & __a) [inline]`

Copy constructor with alternative allocator.

Definition at line 341 of file stl_vector.h.

5.1032.2.8 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::vector<_Tp, _Alloc>::vector (vector<_Tp, _Alloc> && __rv, const allocator_type & __m) [inline], [noexcept]`

Move constructor with alternative allocator.

Definition at line 350 of file stl_vector.h.

5.1032.2.9 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::vector<_Tp, _Alloc>::vector (initializer_list<value_type> __l, const allocator_type & __a = allocator_type()) [inline]`

Builds a vector from an initializer list.

Parameters

$_l$	An initializer_list.
$_a$	An allocator.

Create a vector consisting of copies of the elements in the initializer_list $_l$.

This will call the element type's copy constructor N times (where N is $_l.size()$) and do no memory reallocation.

Definition at line 375 of file `stl_vector.h`.

```
5.1032.2.10  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> template<typename _InputIterator, typename =
std::_RequireInputIter<_InputIterator>>> std::vector<_Tp, _Alloc>::vector ( _InputIterator __first, _InputIterator
__last, const allocator_type & __a = allocator_type() ) [inline]
```

Builds a vector from a range.

Parameters

$_first$	An input iterator.
$_last$	An input iterator.
$_a$	An allocator.

Create a vector consisting of copies of the elements from $[first, last)$.

If the iterators are forward, bidirectional, or random-access, then this will call the elements' copy constructor N times (where N is `distance(first, last)`) and do no memory reallocation. But if only input iterators are used, then this will do at most 2N calls to the copy constructor, and logN memory reallocations.

Definition at line 403 of file `stl_vector.h`.

```
5.1032.2.11  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> std::vector<_Tp, _Alloc>::~~vector ( )
[inline], [noexcept]
```

The dtor only erases the elements, and note that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 425 of file `stl_vector.h`.

5.1032.3 Member Function Documentation

```
5.1032.3.1  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> template<typename _ForwardIterator> pointer
std::vector<_Tp, _Alloc>::_M_allocate_and_copy ( size_type __n, _ForwardIterator __first, _ForwardIterator __last
) [inline], [protected]
```

Memory expansion handler. Uses the member allocation function to obtain n bytes of memory, and then copies $[first, last)$ into it.

Definition at line 1219 of file `stl_vector.h`.

5.1032.3.2 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::vector<_Tp, _Alloc>::M_range_check (size_type __n) const [inline], [protected]`

Safety check used only from at().

Definition at line 801 of file stl_vector.h.

5.1032.3.3 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::vector<_Tp, _Alloc>::assign (size_type __n, const value_type & __val) [inline]`

Assigns a given value to a vector.

Parameters

<code>__n</code>	Number of elements to be assigned.
<code>__val</code>	Value to be assigned.

This function fills a vector with `__n` copies of the given value. Note that the assignment completely changes the vector and that the resulting vector's size is the same as the number of elements assigned. Old data may be lost.

Definition at line 489 of file stl_vector.h.

5.1032.3.4 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> template<typename _InputIterator, typename = std::RequireInputIter<_InputIterator>> void std::vector<_Tp, _Alloc>::assign (_InputIterator __first, _InputIterator __last) [inline]`

Assigns a range to a vector.

Parameters

<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

This function fills a vector with copies of the elements in the range `[__first, __last)`.

Note that the assignment completely changes the vector and that the resulting vector's size is the same as the number of elements assigned. Old data may be lost.

Definition at line 508 of file stl_vector.h.

5.1032.3.5 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::vector<_Tp, _Alloc>::assign (initializer_list<value_type> __l) [inline]`

Assigns an initializer list to a vector.

Parameters

\leftrightarrow	An initializer_list.
$_ \leftrightarrow$	
\leftrightarrow	
$_ \leftrightarrow$	
$_ l$	

This function fills a vector with copies of the elements in the initializer list $_ l$.

Note that the assignment completely changes the vector and that the resulting vector's size is the same as the number of elements assigned. Old data may be lost.

Definition at line 534 of file `stl_vector.h`.

```
5.1032.3.6  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> reference std::vector<_Tp, _Alloc>::at (
              size_type __n ) [inline]
```

Provides access to the data contained in the vector.

Parameters

$_ \leftrightarrow$	The index of the element for which data should be accessed.
$_ n$	

Returns

Read/write reference to data.

Exceptions

<i>std::out_of_range</i>	If $_ n$ is an invalid index.
--------------------------	--------------------------------

This function provides for safer data access. The parameter is first checked that it is in the range of the vector. The function throws `out_of_range` if the check fails.

Definition at line 823 of file `stl_vector.h`.

```
5.1032.3.7  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reference std::vector<_Tp, _Alloc>::at (
              size_type __n ) const [inline]
```

Provides access to the data contained in the vector.

Parameters

$_ \leftrightarrow$	The index of the element for which data should be accessed.
$_ n$	

Returns

Read-only (constant) reference to data.

Exceptions

<code>std::out_of_range</code>	If <code>__n</code> is an invalid index.
--------------------------------	--

This function provides for safer data access. The parameter is first checked that it is in the range of the vector. The function throws `out_of_range` if the check fails.

Definition at line 841 of file `stl_vector.h`.

5.1032.3.8 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> reference std::vector<_Tp, _Alloc>::back ()`
`[inline], [noexcept]`

Returns a read/write reference to the data at the last element of the vector.

Definition at line 868 of file `stl_vector.h`.

Referenced by `std::piecewise_linear_distribution<_RealType>::max()`.

5.1032.3.9 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reference std::vector<_Tp, _Alloc>::back () const` `[inline], [noexcept]`

Returns a read-only (constant) reference to the data at the last element of the vector.

Definition at line 876 of file `stl_vector.h`.

5.1032.3.10 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> iterator std::vector<_Tp, _Alloc>::begin ()`
`[inline], [noexcept]`

Returns a read/write iterator that points to the first element in the vector. Iteration is done in ordinary element order.

Definition at line 548 of file `stl_vector.h`.

Referenced by `std::vector<_Tp, _Alloc>::emplace()`, `__gnu_parallel::multiseq_partition()`, `__gnu_parallel::multiseq←_selection()`, `__gnu_parallel::multiway_merge_exact_splitting()`, `std::vector<_Tp, _Alloc>::operator=()`, `std←::operator=()`, `std::operator>>()`, and `std::vector< sub_match<_Bi_iter>, allocator< sub_match<_Bi_iter>> >::vector()`.

5.1032.3.11 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_iterator std::vector<_Tp, _Alloc>::begin () const` `[inline], [noexcept]`

Returns a read-only (constant) iterator that points to the first element in the vector. Iteration is done in ordinary element order.

Definition at line 557 of file `stl_vector.h`.

5.1032.3.12 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> size_type std::vector<_Tp, _Alloc>::capacity () const [inline], [noexcept]`

Returns the total number of elements that the vector can hold before needing to allocate more memory.

Definition at line 735 of file `stl_vector.h`.

Referenced by `std::vector<_Tp, _Alloc>::emplace()`.

5.1032.3.13 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_iterator std::vector<_Tp, _Alloc>::cbegin () const [inline], [noexcept]`

Returns a read-only (constant) iterator that points to the first element in the vector. Iteration is done in ordinary element order.

Definition at line 621 of file `stl_vector.h`.

5.1032.3.14 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_iterator std::vector<_Tp, _Alloc>::cend () const [inline], [noexcept]`

Returns a read-only (constant) iterator that points one past the last element in the vector. Iteration is done in ordinary element order.

Definition at line 630 of file `stl_vector.h`.

5.1032.3.15 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::vector<_Tp, _Alloc>::clear () [inline], [noexcept]`

Erases all the elements. Note that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1209 of file `stl_vector.h`.

5.1032.3.16 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reverse_iterator std::vector<_Tp, _Alloc>::crbegin () const [inline], [noexcept]`

Returns a read-only (constant) reverse iterator that points to the last element in the vector. Iteration is done in reverse element order.

Definition at line 639 of file `stl_vector.h`.

5.1032.3.17 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reverse_iterator std::vector<_Tp, _Alloc>::crend () const [inline], [noexcept]`

Returns a read-only (constant) reverse iterator that points to one before the first element in the vector. Iteration is done in reverse element order.

Definition at line 648 of file `stl_vector.h`.

5.1032.3.18 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> _Tp* std::vector<_Tp, _Alloc>::data () [inline], [noexcept]`

Returns a pointer such that `[data(), data() + size())` is a valid range. For a non-empty vector, `data() == &front()`.

Definition at line 891 of file `stl_vector.h`.

Referenced by `std::regex_traits<_Ch_type>::transform_primary()`.

5.1032.3.19 `template<typename _Tp, typename _Alloc> template<typename... _Args> vector<_Tp, _Alloc>::iterator vector::emplace (const_iterator __position, _Args &&... __args)`

Inserts an object in vector before specified iterator.

Parameters

<code>__position</code>	A const_iterator into the vector.
<code>__args</code>	Arguments.

Returns

An iterator that points to the inserted data.

This function will insert an object of type T constructed with `T(std::forward<Args>(args)...) before the specified location.` Note that this kind of operation could be expensive for a vector and if it is frequently used the user should consider using `std::list`.

Definition at line 302 of file `vector.tcc`.

References `std::__addressof()`, `std::_Destroy()`, `std::advance()`, `std::begin()`, `std::vector< _Tp, _Alloc >::begin()`, `std::vector< _Tp, _Alloc >::capacity()`, `std::cbegin()`, `std::distance()`, `std::end()`, `std::vector< _Tp, _Alloc >::end()`, and `std::vector< _Tp, _Alloc >::size()`.

Referenced by `std::vector< _Tp, _Alloc >::operator=()`.

5.1032.3.20 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> bool std::vector< _Tp, _Alloc >::empty ()`
`const [inline], [noexcept]`

Returns true if the vector is empty. (Thus `begin()` would equal `end()`.)

Definition at line 744 of file `stl_vector.h`.

Referenced by `std::piecewise_linear_distribution< _RealType >::densities()`, `std::piecewise_linear_distribution< _RealType >::intervals()`, `std::piecewise_linear_distribution< _RealType >::max()`, and `std::piecewise_linear_distribution< _RealType >::min()`.

5.1032.3.21 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> iterator std::vector< _Tp, _Alloc >::end ()`
`[inline], [noexcept]`

Returns a read/write iterator that points one past the last element in the vector. Iteration is done in ordinary element order.

Definition at line 566 of file `stl_vector.h`.

Referenced by `std::vector< _Tp, _Alloc >::emplace()`, `__gnu_parallel::multiseq_partition()`, `__gnu_parallel::multiseq_selection()`, `__gnu_parallel::multiway_merge_exact_splitting()`, `std::vector< _Tp, _Alloc >::operator=()`, `std::vector< _Tp, _Alloc >::operator==()`, `std::operator>>()`, and `std::vector< sub_match< _Bi_iter >, allocator< sub_match< _Bi_iter > >::vector()`.

5.1032.3.22 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_iterator std::vector< _Tp, _Alloc >::end ()`
`const [inline], [noexcept]`

Returns a read-only (constant) iterator that points one past the last element in the vector. Iteration is done in ordinary element order.

Definition at line 575 of file `stl_vector.h`.

5.1032.3.23 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> iterator std::vector< _Tp, _Alloc >::erase (`
`const_iterator __position) [inline]`

Remove element at given position.

Parameters

<code>__position</code>	Iterator pointing to element to be erased.
-------------------------	--

Returns

An iterator pointing to the next element (or end()).

This function will erase the element at the given position and thus shorten the vector by one.

Note This operation could be expensive and if it is frequently used the user should consider using `std::list`. The user is also cautioned that this function only erases the element, and that if the element is itself a pointer, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1147 of file `stl_vector.h`.

```
5.1032.3.24  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> iterator std::vector<_Tp, _Alloc>::erase (
               const_iterator __first, const_iterator __last ) [inline]
```

Remove a range of elements.

Parameters

<code>__first</code>	Iterator pointing to the first element to be erased.
<code>__last</code>	Iterator pointing to one past the last element to be erased.

Returns

An iterator pointing to the element pointed to by `__last` prior to erasing (or end()).

This function will erase the elements in the range `[__first,__last)` and shorten the vector accordingly.

Note This operation could be expensive and if it is frequently used the user should consider using `std::list`. The user is also cautioned that this function only erases the elements, and that if the elements themselves are pointers, the pointed-to memory is not touched in any way. Managing the pointer is the user's responsibility.

Definition at line 1174 of file `stl_vector.h`.

```
5.1032.3.25  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> reference std::vector<_Tp, _Alloc>::front ( )
               [inline], [noexcept]
```

Returns a read/write reference to the data at the first element of the vector.

Definition at line 852 of file `stl_vector.h`.

Referenced by `std::piecewise_linear_distribution<_RealType>::min()`.

5.1032.3.26 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reference std::vector<_Tp, _Alloc>::front () const [inline], [noexcept]`

Returns a read-only (constant) reference to the data at the first element of the vector.

Definition at line 860 of file `stl_vector.h`.

5.1032.3.27 `template<typename _Tp, typename _Alloc > vector<_Tp, _Alloc>::iterator vector::insert (const_iterator __position, const value_type & __x)`

Inserts given value into vector before specified iterator.

Parameters

<code>__position</code>	A <code>const_iterator</code> into the vector.
<code>__x</code>	Data to be inserted.

Returns

An iterator that points to the inserted data.

This function will insert a copy of the given value before the specified location. Note that this kind of operation could be expensive for a vector and if it is frequently used the user should consider using `std::list`.

Definition at line 109 of file `vector.tcc`.

References `std::begin()`, `std::cbegin()`, `std::end()`, and `std::vector<_Tp, _Alloc>::operator=()`.

5.1032.3.28 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> iterator std::vector<_Tp, _Alloc>::insert (const_iterator __position, value_type && __x) [inline]`

Inserts given rvalue into vector before specified iterator.

Parameters

<code>__position</code>	A <code>const_iterator</code> into the vector.
<code>__x</code>	Data to be inserted.

Returns

An iterator that points to the inserted data.

This function will insert a copy of the given rvalue before the specified location. Note that this kind of operation could be expensive for a vector and if it is frequently used the user should consider using `std::list`.

Definition at line 1015 of file `stl_vector.h`.

5.1032.3.29 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> iterator std::vector<_Tp, _Alloc>::insert (const_iterator __position, initializer_list< value_type > __l) [inline]`

Inserts an initializer_list into the vector.

Parameters

<code>__position</code>	An iterator into the vector.
<code>__l</code>	An initializer_list.

This function will insert copies of the data in the initializer_list *l* into the vector before the location specified by *position*.

Note that this kind of operation could be expensive for a vector and if it is frequently used the user should consider using `std::list`.

Definition at line 1032 of file `stl_vector.h`.

5.1032.3.30 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> iterator std::vector<_Tp, _Alloc>::insert (const_iterator __position, size_type __n, const value_type & __x) [inline]`

Inserts a number of copies of given data into the vector.

Parameters

<code>__position</code>	A const_iterator into the vector.
<code>__n</code>	Number of elements to be inserted.
<code>__x</code>	Data to be inserted.

Returns

An iterator that points to the inserted data.

This function will insert a specified number of copies of the given data before the location specified by *position*.

Note that this kind of operation could be expensive for a vector and if it is frequently used the user should consider using `std::list`.

Definition at line 1052 of file `stl_vector.h`.

5.1032.3.31 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>> iterator std::vector<_Tp, _Alloc>::insert (const_iterator __position, _InputIterator __first, _InputIterator __last) [inline]`

Inserts a range into the vector.

Parameters

<code>__position</code>	A const_iterator into the vector.
<code>__first</code>	An input iterator.
<code>__last</code>	An input iterator.

Returns

An iterator that points to the inserted data.

This function will insert copies of the data in the range [`__first`,`__last`) into the vector before the location specified by *pos*.

Note that this kind of operation could be expensive for a vector and if it is frequently used the user should consider using `std::list`.

Definition at line 1096 of file `stl_vector.h`.

```
5.1032.3.32  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> size_type std::vector<_Tp, _Alloc>::max_size
              ( ) const    [inline], [noexcept]
```

Returns the `size()` of the largest possible vector.

Definition at line 660 of file `stl_vector.h`.

```
5.1032.3.33  template<typename _Tp, typename _Alloc> vector<_Tp, _Alloc> & vector::operator= ( const vector<_Tp,
              _Alloc> & __x )
```

Vector assignment operator.

Parameters

<code>__x</code>	A vector of identical element and allocator types.
------------------	--

All the elements of `__x` are copied, but any extra memory in `__x` (for fast expansion) will not be copied. Unlike the copy constructor, the allocator object is not copied.

Definition at line 168 of file `vector.tcc`.

References `std::_Destroy()`, `std::advance()`, `std::begin()`, `std::vector<_Tp, _Alloc>::begin()`, `std::distance()`, `std::vector<_Tp, _Alloc>::emplace()`, `std::end()`, `std::vector<_Tp, _Alloc>::end()`, `std::fill_n()`, and `std::vector<_Tp, _Alloc>::size()`.

Referenced by `std::vector<_Tp, _Alloc>::insert()`.

```
5.1032.3.34  template<typename _Tp, typename _Alloc = std::allocator<_Tp>> vector& std::vector<_Tp, _Alloc>::operator=
              ( vector<_Tp, _Alloc> && __x )    [inline], [noexcept]
```

Vector move assignment operator.

Parameters

<code>__x</code>	A vector of identical element and allocator types.
------------------	--

The contents of `__x` are moved into this vector (without copying, if the allocators permit it). `__x` is a valid, but unspecified vector.

Definition at line 450 of file `stl_vector.h`.

5.1032.3.35 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> vector& std::vector<_Tp, _Alloc>::operator=(initializer_list< value_type > __l) [inline]`

Vector list assignment operator.

Parameters

<code>↔</code>	An <code>initializer_list</code> .
<code>__↔</code>	
<code>↔</code>	
<code>__↔</code>	
<code>l</code>	

This function fills a vector with copies of the elements in the initializer list `__l`.

Note that the assignment completely changes the vector and that the resulting vector's size is the same as the number of elements assigned. Old data may be lost.

Definition at line 471 of file `stl_vector.h`.

5.1032.3.36 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> reference std::vector<_Tp, _Alloc>::operator[](size_type __n) [inline], [noexcept]`

Subscript access to the data contained in the vector.

Parameters

<code>__↔</code>	The index of the element for which data should be accessed.
<code>__n</code>	

Returns

Read/write reference to data.

This operator allows for easy, array-style, data access. Note that data access with this operator is unchecked and `out_of_range` lookups are not defined. (For checked lookups see `at()`.)

Definition at line 780 of file `stl_vector.h`.

5.1032.3.37 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reference std::vector<_Tp, _Alloc>::operator[](size_type __n) const [inline], [noexcept]`

Subscript access to the data contained in the vector.

Parameters

$_n$	The index of the element for which data should be accessed.
-------	---

Returns

Read-only (constant) reference to data.

This operator allows for easy, array-style, data access. Note that data access with this operator is unchecked and out_of_range lookups are not defined. (For checked lookups see at().)

Definition at line 795 of file stl_vector.h.

5.1032.3.38 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::vector<_Tp, _Alloc>::pop_back ()`
`[inline], [noexcept]`

Removes last element.

This is a typical stack operation. It shrinks the vector by one.

Note that no data is returned, and if the last element's data is needed, it should be retrieved before pop_back() is called.

Definition at line 950 of file stl_vector.h.

5.1032.3.39 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::vector<_Tp, _Alloc>::push_back (`
`const value_type &__x) [inline]`

Add data to the end of the vector.

Parameters

$_x$	Data to be added.
-------	-------------------

This is a typical stack operation. The function creates an element at the end of the vector and assigns the given data to it. Due to the nature of a vector this operation can be done in constant time if the vector has preallocated space available.

Definition at line 914 of file stl_vector.h.

Referenced by `__gnu_parallel::multiseq_partition()`, `__gnu_parallel::multiseq_selection()`, and `std::operator>>()`.

5.1032.3.40 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> reverse_iterator std::vector<_Tp, _Alloc>`
`>::rbegin () [inline], [noexcept]`

Returns a read/write reverse iterator that points to the last element in the vector. Iteration is done in reverse element order.

Definition at line 584 of file stl_vector.h.

5.1032.3.41 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reverse_iterator std::vector<_Tp, _Alloc>::rbegin () const [inline], [noexcept]`

Returns a read-only (constant) reverse iterator that points to the last element in the vector. Iteration is done in reverse element order.

Definition at line 593 of file `stl_vector.h`.

5.1032.3.42 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> reverse_iterator std::vector<_Tp, _Alloc>::rend () [inline], [noexcept]`

Returns a read/write reverse iterator that points to one before the first element in the vector. Iteration is done in reverse element order.

Definition at line 602 of file `stl_vector.h`.

5.1032.3.43 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> const_reverse_iterator std::vector<_Tp, _Alloc>::rend () const [inline], [noexcept]`

Returns a read-only (constant) reverse iterator that points to one before the first element in the vector. Iteration is done in reverse element order.

Definition at line 611 of file `stl_vector.h`.

5.1032.3.44 `template<typename _Tp, typename _Alloc> void vector::reserve (size_type __n)`

Attempt to preallocate enough memory for specified number of elements.

Parameters

<code>__n</code>	Number of elements required.
------------------	------------------------------

Exceptions

<code>std::length_error</code>	If <code>n</code> exceeds <code>max_size()</code> .
--------------------------------	---

This function attempts to reserve enough memory for the vector to hold the specified number of elements. If the number requested is more than `max_size()`, `length_error` is thrown.

The advantage of this function is that if optimal code is a necessity and the user can determine the number of elements that will be required, the user can reserve the memory in advance, and thus prevent a possible reallocation of memory and copying of vector data.

Definition at line 66 of file `vector.tcc`.

References `std::_Destroy()`.

Referenced by `std::operator>>()`.

5.1032.3.45 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::vector<_Tp, _Alloc>::resize (size_type __new_size) [inline]`

Resizes the vector to the specified number of elements.

Parameters

<code>__new_size</code>	Number of elements the vector should contain.
-------------------------	---

This function will resize the vector to the specified number of elements. If the number is smaller than the vector's current size the vector is truncated, otherwise default constructed elements are appended.

Definition at line 674 of file `stl_vector.h`.

Referenced by `__gnu_parallel::__shrink_and_double()`, and `__gnu_parallel::multiway_merge_exact_splitting()`.

5.1032.3.46 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::vector<_Tp, _Alloc>::resize (size_type __new_size, const value_type & __x) [inline]`

Resizes the vector to the specified number of elements.

Parameters

<code>__new_size</code>	Number of elements the vector should contain.
<code>__x</code>	Data with which new elements should be populated.

This function will resize the vector to the specified number of elements. If the number is smaller than the vector's current size the vector is truncated, otherwise the vector is extended and new elements are populated with given data.

Definition at line 694 of file `stl_vector.h`.

5.1032.3.47 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::vector<_Tp, _Alloc>::shrink_to_fit () [inline]`

A non-binding request to reduce capacity() to size().

Definition at line 726 of file `stl_vector.h`.

5.1032.3.48 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> size_type std::vector<_Tp, _Alloc>::size () const [inline], [noexcept]`

Returns the number of elements in the vector.

Definition at line 655 of file `stl_vector.h`.

Referenced by `__gnu_parallel::__shrink()`, `__gnu_parallel::__shrink_and_double()`, `std::vector<_Tp, _Alloc>::emplace()`, `__gnu_parallel::list_partition()`, `std::vector<_Tp, _Alloc>::operator=()`, `std::operator==(, std::vector<_Tp, _Alloc>)`, and `std::regex_traits<_Ch_type>::transform_primary()`.

5.1032.3.49 `template<typename _Tp, typename _Alloc = std::allocator<_Tp>> void std::vector<_Tp, _Alloc>::swap (vector<_Tp, _Alloc> & __x) [inline], [noexcept]`

Swaps data with another vector.

Parameters

<code>_↔_X</code>	A vector of the same element and allocator types.
-------------------	---

This exchanges the elements between two vectors in constant time. (Three pointers, so it should be quite fast.) Note that the global `std::swap()` function is specialized such that `std::swap(v1,v2)` will feed to this function.

Definition at line 1195 of file `stl_vector.h`.

The documentation for this class was generated from the following files:

- [stl_vector.h](#)
- [vector.tcc](#)

5.1033 `std::vector< bool, _Alloc >` Class Template Reference

Inherits `std::_Bvector_base< _Alloc >`.

Public Types

- typedef `_Alloc` **allocator_type**
- typedef `_Bit_const_iterator` **const_iterator**
- typedef `const bool *` **const_pointer**
- typedef `bool` **const_reference**
- typedef `std::reverse_iterator< const_iterator >` **const_reverse_iterator**
- typedef `ptrdiff_t` **difference_type**
- typedef `_Bit_iterator` **iterator**
- typedef `_Bit_reference *` **pointer**
- typedef `_Bit_reference` **reference**
- typedef `std::reverse_iterator< iterator >` **reverse_iterator**
- typedef `size_t` **size_type**
- typedef `bool` **value_type**

Public Member Functions

- **vector** (`const allocator_type &__a`)
- **vector** (`size_type __n, const allocator_type &__a=allocator_type()`)
- **vector** (`size_type __n, const bool &__value, const allocator_type &__a=allocator_type()`)
- **vector** (`const vector &__x`)
- **vector** (`vector &&__x`) `noexcept`
- **vector** (`vector &&__x, const allocator_type &__a`) `noexcept(_Bit_alloc_traits::S_always_equal())`
- **vector** (`const vector &__x, const allocator_type &__a`)
- **vector** (`initializer_list< bool > __l, const allocator_type &__a=allocator_type()`)
- template<typename `_InputIterator` , typename = `std::RequireInputIter<_InputIterator>>`
vector (`_InputIterator __first, _InputIterator __last, const allocator_type &__a=allocator_type()`)
- void **assign** (`size_type __n, const bool &__x`)

- template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>>
void **assign** (_InputIterator __first, _InputIterator __last)
- void **assign** (initializer_list< bool > __l)
- reference **at** (size_type __n)
- const_reference **at** (size_type __n) const
- reference **back** ()
- const_reference **back** () const
- iterator **begin** () noexcept
- const_iterator **begin** () const noexcept
- size_type **capacity** () const noexcept
- const_iterator **cbegin** () const noexcept
- const_iterator **cend** () const noexcept
- void **clear** () noexcept
- const_reverse_iterator **crbegin** () const noexcept
- const_reverse_iterator **crend** () const noexcept
- void **data** () noexcept
- template<typename... _Args>
iterator **emplace** (const_iterator __pos, _Args &&... __args)
- template<typename... _Args>
void **emplace_back** (_Args &&... __args)
- bool **empty** () const noexcept
- iterator **end** () noexcept
- const_iterator **end** () const noexcept
- iterator **erase** (const_iterator __position)
- iterator **erase** (const_iterator __first, const_iterator __last)
- void **flip** () noexcept
- reference **front** ()
- const_reference **front** () const
- allocator_type **get_allocator** () const
- iterator **insert** (const_iterator __position, const bool &__x=bool())
- template<typename _InputIterator, typename = std::_RequireInputIter<_InputIterator>>>
iterator **insert** (const_iterator __position, _InputIterator __first, _InputIterator __last)
- iterator **insert** (const_iterator __position, size_type __n, const bool &__x)
- iterator **insert** (const_iterator __p, initializer_list< bool > __l)
- size_type **max_size** () const noexcept
- vector & **operator=** (const vector &__x)
- vector & **operator=** (vector &&__x) noexcept(_Bit_alloc_traits::_S_nothrow_move())
- vector & **operator=** (initializer_list< bool > __l)
- reference **operator[]** (size_type __n)
- const_reference **operator[]** (size_type __n) const
- void **pop_back** ()
- void **push_back** (bool __x)
- reverse_iterator **rbegin** () noexcept
- const_reverse_iterator **rbegin** () const noexcept
- reverse_iterator **rend** () noexcept
- const_reverse_iterator **rend** () const noexcept
- void **reserve** (size_type __n)
- void **resize** (size_type __new_size, bool __x=bool())
- void **shrink_to_fit** ()
- size_type **size** () const noexcept
- void **swap** (vector &__x) noexcept

Static Public Member Functions

- static void **swap** (reference __x, reference __y) noexcept

Protected Types

- typedef [__gnu_cxx::__alloc_traits](#)< _Alloc >::template rebind< _Bit_type >::other **_Bit_alloc_type**

Protected Member Functions

- **_Bit_pointer _M_allocate** (size_t __n)
- template<typename _InputIterator >
void **_M_assign_aux** (_InputIterator __first, _InputIterator __last, [std::input_iterator_tag](#))
- template<typename _ForwardIterator >
void **_M_assign_aux** (_ForwardIterator __first, _ForwardIterator __last, [std::forward_iterator_tag](#))
- template<typename _Integer >
void **_M_assign_dispatch** (_Integer __n, _Integer __val, __true_type)
- template<class _InputIterator >
void **_M_assign_dispatch** (_InputIterator __first, _InputIterator __last, __false_type)
- size_type **_M_check_len** (size_type __n, const char *__s) const
- iterator **_M_copy_aligned** (const_iterator __first, const_iterator __last, iterator __result)
- void **_M_deallocate** ()
- iterator **_M_erase** (iterator __pos)
- iterator **_M_erase** (iterator __first, iterator __last)
- void **_M_erase_at_end** (iterator __pos)
- void **_M_fill_assign** (size_t __n, bool __x)
- void **_M_fill_insert** (iterator __position, size_type __n, bool __x)
- **_Bit_alloc_type & _M_get_Bit_allocator** () noexcept
- const **_Bit_alloc_type & _M_get_Bit_allocator** () const noexcept
- void **_M_initialize** (size_type __n)
- template<typename _Integer >
void **_M_initialize_dispatch** (_Integer __n, _Integer __x, __true_type)
- template<typename _InputIterator >
void **_M_initialize_dispatch** (_InputIterator __first, _InputIterator __last, __false_type)
- template<typename _InputIterator >
void **_M_initialize_range** (_InputIterator __first, _InputIterator __last, [std::input_iterator_tag](#))
- template<typename _ForwardIterator >
void **_M_initialize_range** (_ForwardIterator __first, _ForwardIterator __last, [std::forward_iterator_tag](#))
- void **_M_insert_aux** (iterator __position, bool __x)
- template<typename _Integer >
void **_M_insert_dispatch** (iterator __pos, _Integer __n, _Integer __x, __true_type)
- template<typename _InputIterator >
void **_M_insert_dispatch** (iterator __pos, _InputIterator __first, _InputIterator __last, __false_type)
- template<typename _InputIterator >
void **_M_insert_range** (iterator __pos, _InputIterator __first, _InputIterator __last, [std::input_iterator_tag](#))
- template<typename _ForwardIterator >
void **_M_insert_range** (iterator __position, _ForwardIterator __first, _ForwardIterator __last, [std::forward_iterator_tag](#))
- void **_M_range_check** (size_type __n) const
- void **_M_reallocate** (size_type __n)
- bool **_M_shrink_to_fit** ()

Static Protected Member Functions

- `static size_t S_nword (size_t __n)`

Protected Attributes

- `_Bvector_impl M_impl`

Friends

- `template<typename >`
`struct hash`

5.1033.1 Detailed Description

```
template<typename _Alloc>  
class std::vector< bool, _Alloc >
```

A specialization of vector for booleans which offers fixed time access to individual elements in any order.

Template Parameters

<code>_Alloc</code>	Allocator type.
---------------------	-----------------

Note that `vector<bool>` does not actually meet the requirements for being a container. This is because the reference and pointer types are not really references and pointers to `bool`. See DR96 for details.

See also

[vector](#) for function documentation.

In some terminology a vector can be described as a dynamic C-style array, it offers fast and efficient access to individual elements in any order and saves the user from worrying about memory and size allocation. Subscripting (`[]`) access is also provided as with C-style arrays.

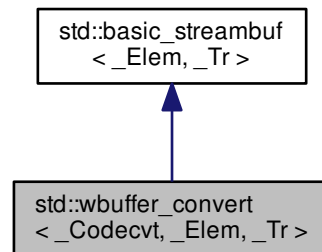
Definition at line 541 of file `stl_bvector.h`.

The documentation for this class was generated from the following files:

- [stl_bvector.h](#)
- [vector.tcc](#)

5.1034 `std::wbuffer_convert<_Codecvt,_Elem,_Tr>` Class Template Reference

Inheritance diagram for `std::wbuffer_convert<_Codecvt,_Elem,_Tr>`:



Public Types

- `typedef _Codecvt::state_type` **state_type**
- `typedef _Elem` **char_type**
- `typedef _Tr` **traits_type**
- `typedef traits_type::int_type` **int_type**
- `typedef traits_type::pos_type` **pos_type**
- `typedef traits_type::off_type` **off_type**
- `typedef basic_streambuf<char_type, traits_type>` **__streambuf_type**

Public Member Functions

- `wbuffer_convert` (`streambuf * __bytebuf=0, _Codecvt * __pcvt=new _Codecvt, state_type __state=state_type()`)
- `wbuffer_convert` (`const wbuffer_convert &)=delete`
- `locale getloc` () const
- `streamsize in_avail` ()
- `wbuffer_convert & operator=` (`const wbuffer_convert &)=delete`
- `locale pubimbue` (`const locale & __loc`)
- `streambuf * rdbuf` () const noexcept
- `streambuf * rdbuf` (`streambuf * __bytebuf`) noexcept
- `int_type sbumpc` ()
- `int_type sgetc` ()
- `streamsize sgetn` (`char_type * __s, streamsize __n`)
- `int_type snextc` ()
- `int_type sputbackc` (`char_type __c`)
- `int_type sputc` (`char_type __c`)

- [streamsize sputn](#) (const [char_type](#) *__s, [streamsize](#) __n)
- [state_type state](#) () const noexcept
- [int_type sungetc](#) ()
- [basic_streambuf](#) * [pubsetbuf](#) ([char_type](#) *__s, [streamsize](#) __n)
- [pos_type](#) [pubseekoff](#) ([off_type](#) __off, [ios_base::seekdir](#) __way, [ios_base::openmode](#) __mode=[ios_base::in](#)|[ios_base::out](#))
- [pos_type](#) [pubseekpos](#) ([pos_type](#) __sp, [ios_base::openmode](#) __mode=[ios_base::in](#)|[ios_base::out](#))
- [int](#) [pubsync](#) ()

Protected Member Functions

- void [__safe_gbump](#) ([streamsize](#) __n)
- void [__safe_pbump](#) ([streamsize](#) __n)
- void [gbump](#) (int __n)
- virtual void [imbue](#) (const [locale](#) &__loc)
- [_Wide_streambuf::int_type](#) [overflow](#) (typename [_Wide_streambuf::int_type](#) __out)
- virtual [int_type](#) [pbackfail](#) ([int_type](#) __c=[traits_type::eof](#)())
- void [pbump](#) (int __n)
- virtual [pos_type](#) [seekoff](#) ([off_type](#), [ios_base::seekdir](#), [ios_base::openmode](#)=[ios_base::in](#)|[ios_base::out](#))
- virtual [pos_type](#) [seekpos](#) ([pos_type](#), [ios_base::openmode](#)=[ios_base::in](#)|[ios_base::out](#))
- virtual [basic_streambuf< char_type, _Tr > * setbuf](#) ([char_type](#) *, [streamsize](#))
- void [setg](#) ([char_type](#) *__gbeg, [char_type](#) *__gnext, [char_type](#) *__gend)
- void [setp](#) ([char_type](#) *__pbeg, [char_type](#) *__pend)
- virtual [streamsize](#) [showmanyc](#) ()
- void [swap](#) ([basic_streambuf](#) &__sb)
- [int](#) [sync](#) ()
- virtual [int_type](#) [uflow](#) ()
- [_Wide_streambuf::int_type](#) [underflow](#) ()
- virtual [streamsize](#) [xsgetn](#) ([char_type](#) *__s, [streamsize](#) __n)
- [streamsize](#) [xsputn](#) (const typename [_Wide_streambuf::char_type](#) *__s, [streamsize](#) __n)
- virtual [streamsize](#) [xspn](#) (const [char_type](#) *__s, [streamsize](#) __n)
- [char_type](#) * [eback](#) () const
- [char_type](#) * [gptr](#) () const
- [char_type](#) * [egptr](#) () const
- [char_type](#) * [pbase](#) () const
- [char_type](#) * [pptr](#) () const
- [char_type](#) * [epptr](#) () const

Protected Attributes

- [locale](#) [_M_buf_locale](#)
- [char_type](#) * [_M_in_beg](#)
- [char_type](#) * [_M_in_cur](#)
- [char_type](#) * [_M_in_end](#)
- [char_type](#) * [_M_out_beg](#)
- [char_type](#) * [_M_out_cur](#)
- [char_type](#) * [_M_out_end](#)

5.1034.1 Detailed Description

```
template<typename _Codecvt, typename _Elem = wchar_t, typename _Tr = char_traits<_Elem>>
class std::wbuffer_convert< _Codecvt, _Elem, _Tr >
```

Buffer conversions.

Definition at line 318 of file locale_conv.h.

5.1034.2 Member Typedef Documentation

5.1034.2.1 `typedef basic_streambuf<char_type, traits_type> std::basic_streambuf<_Elem, _Tr>::__streambuf_type` [inherited]

This is a non-standard type.

Definition at line 138 of file streambuf.

5.1034.2.2 `typedef _Elem std::basic_streambuf<_Elem, _Tr>::char_type` [inherited]

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

Definition at line 129 of file streambuf.

5.1034.2.3 `typedef traits_type::int_type std::basic_streambuf<_Elem, _Tr>::int_type` [inherited]

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

Definition at line 131 of file streambuf.

5.1034.2.4 `typedef traits_type::off_type std::basic_streambuf<_Elem, _Tr>::off_type` [inherited]

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

Definition at line 133 of file streambuf.

5.1034.2.5 `typedef traits_type::pos_type std::basic_streambuf<_Elem, _Tr>::pos_type` [inherited]

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

Definition at line 132 of file streambuf.

5.1034.2.6 `typedef _Tr std::basic_streambuf<_Elem, _Tr>::traits_type` [inherited]

These are standard types. They permit a standardized way of referring to names of (or names dependent on) the template parameters, which are specific to the implementation.

Definition at line 130 of file streambuf.

5.1034.3 Constructor & Destructor Documentation

5.1034.3.1 `template<typename _Codecvt, typename _Elem = wchar_t, typename _Tr = char_traits<_Elem>>
std::wbuffer_convert< _Codecvt, _Elem, _Tr >::wbuffer_convert (streambuf * __bytebuf = 0, _Codecvt *
__pcvt = new _Codecvt, state_type __state = state_type())` [inline], [explicit]

Default constructor.

Parameters

<code>__bytebuf</code>	The underlying byte stream buffer.
<code>__pcvt</code>	The facet to use for conversions.
<code>__state</code>	Initial conversion state.

Takes ownership of `__pcvt` and will delete it in the destructor.

Definition at line 334 of file `locale_conv.h`.

5.1034.4 Member Function Documentation

5.1034.4.1 `char_type* std::basic_streambuf<_Elem, _Tr>::eback() const` `[inline]`, `[protected]`, `[inherited]`

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Definition at line 482 of file `streambuf`.

5.1034.4.2 `char_type* std::basic_streambuf<_Elem, _Tr>::egptr() const` `[inline]`, `[protected]`, `[inherited]`

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Definition at line 488 of file `streambuf`.

5.1034.4.3 `char_type* std::basic_streambuf<_Elem, _Tr>::eptr() const` `[inline]`, `[protected]`, `[inherited]`

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `eptr()` returns the end pointer for the output sequence

Definition at line 535 of file `streambuf`.

5.1034.4.4 `void std::basic_streambuf<_Elem, _Tr>::gbump(int __n)` `[inline]`, `[protected]`, `[inherited]`

Moving the read position.

Parameters

<code>_↔</code>	The delta by which to move.
<code>_n</code>	

This just advances the read position without returning any data.

Definition at line 498 of file `streambuf`.

5.1034.4.5 `locale std::basic_streambuf<_Elem, _Tr>::getloc () const` `[inline]`, `[inherited]`

Locale access.

Returns

The current locale in effect.

If `pubimbue(loc)` has been called, then the most recent `loc` is returned. Otherwise the global locale in effect at the time of construction is returned.

Definition at line 226 of file `streambuf`.

5.1034.4.6 `char_type* std::basic_streambuf<_Elem, _Tr>::gptr () const` `[inline]`, `[protected]`, `[inherited]`

Access to the get area.

These functions are only available to other protected functions, including derived classes.

- `eback()` returns the beginning pointer for the input sequence
- `gptr()` returns the next pointer for the input sequence
- `egptr()` returns the end pointer for the input sequence

Definition at line 485 of file `streambuf`.

5.1034.4.7 `virtual void std::basic_streambuf<_Elem, _Tr>::imbue (const locale & __loc)` `[inline]`, `[protected]`, `[virtual]`, `[inherited]`

Changes translations.

Parameters

<code>__loc</code>	A new locale.
--------------------	---------------

Translations done during I/O which depend on the current locale are changed by this call. The standard adds, *Between*

invocations of this function a class derived from `streambuf` can safely cache results of calls to locale functions and to members of facets so obtained.

Note

Base class version does nothing.

Definition at line 576 of file `streambuf`.

5.1034.4.8 `streamsize std::basic_streambuf<_Elem, _Tr>::in_avail ()` `[inline]`, `[inherited]`

Looking ahead into the stream.

Returns

The number of characters available.

If a read position is available, returns the number of characters available for reading before the buffer must be refilled. Otherwise returns the derived `showmanyc()`.

Definition at line 284 of file `streambuf`.

5.1034.4.9 `template<typename _Codecvt, typename _Elem = wchar_t, typename _Tr = char_traits<_Elem>> _Wide_streambuf::int_type std::wbuffer_convert<_Codecvt, _Elem, _Tr>::overflow (typename _Wide_streambuf::int_type __c)` `[inline]`, `[protected]`, `[virtual]`

Consumes data from the buffer; writes to the controlled sequence.

Parameters

<code>__c</code>	An additional character to consume.
------------------	-------------------------------------

Returns

`eof()` to indicate failure, something else (usually `__c`, or `not_eof()`)

Informally, this function is called when the output buffer is full (or does not exist, as buffering need not actually be done). If a buffer exists, it is *consumed*, with *some effect* on the controlled sequence. (Typically, the buffer is written out to the sequence verbatim.) In either case, the character `c` is also written out, if `__c` is not `eof()`.

For a formal definition of this function, see a good text such as Langer & Kreft, or [27.5.2.4.5]/3-7.

A functioning output `streambuf` can be created by overriding only this function (no buffer area will be used).

Note

Base class version does nothing, returns `eof()`.

Reimplemented from `std::basic_streambuf<_Elem, _Tr>`.

Definition at line 378 of file `locale_conv.h`.

5.1034.4.10 `virtual int_type std::basic_streambuf<_Elem, _Tr>::pbackfail (int_type __c = traits_type::eof()) [inline], [protected], [virtual], [inherited]`

Tries to back up the input sequence.

Parameters

<code>_↵</code>	The character to be inserted back into the sequence.
<code>_C</code>	

Returns

`eof()` on failure, *some other value* on success

Postcondition

The constraints of `gptr()`, `eback()`, and `pptr()` are the same as for `underflow()`.

Note

Base class version does nothing, returns `eof()`.

Definition at line 724 of file `streambuf`.

5.1034.4.11 `char_type* std::basic_streambuf<_Elem, _Tr>::pbase () const [inline], [protected], [inherited]`

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `eptr()` returns the end pointer for the output sequence

Definition at line 529 of file `streambuf`.

5.1034.4.12 `void std::basic_streambuf<_Elem, _Tr>::pbump (int __n) [inline], [protected], [inherited]`

Moving the write position.

Parameters

<code>_↔</code>	The delta by which to move.
<code>_n</code>	

This just advances the write position without returning any data.

Definition at line 545 of file `streambuf`.

5.1034.4.13 `char_type* std::basic_streambuf<_Elem, _Tr>::pptr () const` `[inline]`, `[protected]`, `[inherited]`

Access to the put area.

These functions are only available to other protected functions, including derived classes.

- `pbase()` returns the beginning pointer for the output sequence
- `pptr()` returns the next pointer for the output sequence
- `epptr()` returns the end pointer for the output sequence

Definition at line 532 of file `streambuf`.

5.1034.4.14 `locale std::basic_streambuf<_Elem, _Tr>::pubimbue (const locale & __loc)` `[inline]`, `[inherited]`

Entry point for `imbue()`.

Parameters

<code>__loc</code>	The new locale.
--------------------	-----------------

Returns

The previous locale.

Calls the derived `imbue(__loc)`.

Definition at line 209 of file `streambuf`.

5.1034.4.15 `pos_type std::basic_streambuf<_Elem, _Tr>::pubseekoff (off_type __off, ios_base::seekdir __way, ios_base::openmode __mode = ios_base::in | ios_base::out)` `[inline]`, `[inherited]`

Alters the stream position.

Parameters

<code>__off</code>	Offset.
<code>__way</code>	Value for <code>ios_base::seekdir</code> .
<code>__mode</code>	Value for <code>ios_base::openmode</code> .

Calls virtual `seekoff` function.

Definition at line 251 of file `streambuf`.

5.1034.4.16 `pos_type std::basic_streambuf<_Elem, _Tr>::pubseekpos (pos_type __sp, ios_base::openmode __mode = ios_base::in | ios_base::out) [inline], [inherited]`

Alters the stream position.

Parameters

<code>__sp</code>	Position
<code>__mode</code>	Value for <code>ios_base::openmode</code> .

Calls virtual `seekpos` function.

Definition at line 263 of file `streambuf`.

5.1034.4.17 `basic_streambuf* std::basic_streambuf<_Elem, _Tr>::pubsetbuf (char_type* __s, streamsize __n) [inline], [inherited]`

Entry points for derived buffer functions.

The public versions of `pubfoo` dispatch to the protected derived `foo` member functions, passing the arguments (if any) and returning the result unchanged.

Definition at line 239 of file `streambuf`.

5.1034.4.18 `int std::basic_streambuf<_Elem, _Tr>::pubsync () [inline], [inherited]`

Calls virtual `sync` function.

Definition at line 271 of file `streambuf`.

5.1034.4.19 `int_type std::basic_streambuf<_Elem, _Tr>::sbumpc () [inline], [inherited]`

Getting the next character.

Returns

The next character, or `eof`.

If the input read position is available, returns that character and increments the read pointer, otherwise calls and returns `uflow()`.

Definition at line 316 of file `streambuf`.

5.1034.4.20 `virtual pos_type std::basic_streambuf<_Elem, _Tr>::seekoff (off_type , ios_base::seekdir , ios_base::openmode = ios_base::in | ios_base::out) [inline], [protected], [virtual], [inherited]`

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Definition at line 602 of file `streambuf`.

5.1034.4.21 `virtual pos_type std::basic_streambuf<_Elem, _Tr>::seekpos (pos_type , ios_base::openmode = ios_base::in | ios_base::out) [inline], [protected], [virtual], [inherited]`

Alters the stream positions.

Each derived class provides its own appropriate behavior.

Note

Base class version does nothing, returns a `pos_type` that represents an invalid stream position.

Definition at line 614 of file `streambuf`.

5.1034.4.22 `virtual basic_streambuf<char_type, Tr>* std::basic_streambuf<_Elem, _Tr>::setbuf (char_type* , streamsize) [inline], [protected], [virtual], [inherited]`

Manipulates the buffer.

Each derived class provides its own appropriate behavior. See the next-to-last paragraph of <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html#io.streambuf.buffering> for more on this function.

Note

Base class version does nothing, returns `this`.

Definition at line 591 of file `streambuf`.

5.1034.4.23 `void std::basic_streambuf<_Elem, _Tr>::setg (char_type* __gbeg, char_type* __gnext, char_type* __gend) [inline], [protected], [inherited]`

Setting the three read area pointers.

Parameters

<code>__gbeg</code>	A pointer.
<code>__gnext</code>	A pointer.
<code>__gend</code>	A pointer.

Postcondition

`__gbeg == eback()`, `__gnext == gptr()`, and `__gend == egptr()`

Definition at line 509 of file `streambuf`.

5.1034.4.24 `void std::basic_streambuf<_Elem, _Tr>::setp(char_type* __pbeg, char_type* __pend)` `[inline]`, `[protected]`, `[inherited]`

Setting the three write area pointers.

Parameters

<code>__pbeg</code>	A pointer.
<code>__pend</code>	A pointer.

Postcondition

`__pbeg == pbase()`, `__pbeg == pptr()`, and `__pend == epptr()`

Definition at line 555 of file `streambuf`.

5.1034.4.25 `int_type std::basic_streambuf<_Elem, _Tr>::sgetc()` `[inline]`, `[inherited]`

Getting the next character.

Returns

The next character, or `eof`.

If the input read position is available, returns that character, otherwise calls and returns `underflow()`. Does not move the read position after fetching the character.

Definition at line 338 of file `streambuf`.

5.1034.4.26 `streamsize std::basic_streambuf<_Elem, _Tr>::sgetn(char_type* __s, streamsize __n)` `[inline]`, `[inherited]`

Entry point for `xsgetn`.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	A count.

Returns `xsgetn(__s,__n)`. The effect is to fill `__s[0]` through `__s[__n-1]` with characters from the input sequence, if possible.

Definition at line 357 of file `streambuf`.

5.1034.4.27 `virtual streamsize std::basic_streambuf<_Elem,_Tr>::showmanyc ()` `[inline],[protected],[virtual],[inherited]`

Investigating the data available.

Returns

An estimate of the number of characters available in the input sequence, or -1.

If it returns a positive value, then successive calls to `underflow()` will not return `traits::eof()` until at least that number of characters have been supplied. If `showmanyc()` returns -1, then calls to `underflow()` or `uflow()` will fail. [27.5.2.4.3]/1

Note

Base class version does nothing, returns zero.

The standard adds that *the intention is not only that the calls [to `underflow` or `uflow`] will not return `eof()` but that they will return immediately.*

The standard adds that *the morphemes of `showmanyc` are **es-how-many-see**, not **show-manic**.*

Definition at line 649 of file `streambuf`.

5.1034.4.28 `int_type std::basic_streambuf<_Elem,_Tr>::snextc ()` `[inline],[inherited]`

Getting the next character.

Returns

The next character, or eof.

Calls `sbumpc()`, and if that function returns `traits::eof()`, so does this function. Otherwise, `sgetc()`.

Definition at line 298 of file `streambuf`.

5.1034.4.29 `int_type std::basic_streambuf<_Elem,_Tr>::sputbackc (char_type __c)` `[inline],[inherited]`

Pushing characters back into the input stream.

Parameters

<code>↵</code> <code>_c</code>	The character to push back.
-----------------------------------	-----------------------------

Returns

The previous character, if possible.

Similar to `sungetc()`, but `__c` is pushed onto the stream instead of *the previous character*. If successful, the next character fetched from the input stream will be `__c`.

Definition at line 372 of file `streambuf`.

5.1034.4.30 `int_type std::basic_streambuf<_Elem,_Tr>::sputc (char_type __c)` `[inline],[inherited]`

Entry point for all single-character output functions.

Parameters

<code>↵</code> <code>_c</code>	A character to output.
-----------------------------------	------------------------

Returns

`__c`, if possible.

One of two public output functions.

If a write position is available for the output sequence (i.e., the buffer is not full), stores `__c` in that position, increments the position, and returns `traits::to_int_type (__c)`. If a write position is not available, returns `overflow (↵ __c)`.

Definition at line 424 of file `streambuf`.

5.1034.4.31 `streamsize std::basic_streambuf<_Elem,_Tr>::sputn (const char_type * __s, streamsize __n)`
`[inline],[inherited]`

Entry point for all single-character output functions.

Parameters

<code>↵</code> <code>_s</code>	A buffer read area.
<code>↵</code> <code>_n</code>	A count.

One of two public output functions.

Returns `xspn(__s, __n)`. The effect is to write `__s[0]` through `__s[__n-1]` to the output sequence, if possible.

Definition at line 450 of file `streambuf`.

5.1034.4.32 `template<typename _Codecvt, typename _Elem = wchar_t, typename _Tr = char_traits<_Elem>> state_type
std::wbuffer_convert<_Codecvt, _Elem, _Tr>::state () const [inline], [noexcept]`

The conversion state following the last conversion.

Definition at line 370 of file `locale_conv.h`.

5.1034.4.33 `int_type std::basic_streambuf<_Elem, _Tr>::sungetc () [inline], [inherited]`

Moving backwards in the input stream.

Returns

The previous character, if possible.

If a putback position is available, this function decrements the input pointer and returns that character. Otherwise, calls and returns `pbackfail()`. The effect is to *unget* the last character *gotten*.

Definition at line 397 of file `streambuf`.

5.1034.4.34 `template<typename _Codecvt, typename _Elem = wchar_t, typename _Tr = char_traits<_Elem>> int
std::wbuffer_convert<_Codecvt, _Elem, _Tr>::sync (void) [inline], [protected], [virtual]`

Synchronizes the buffer arrays with the controlled sequences.

Returns

-1 on failure.

Each derived class provides its own appropriate behavior, including the definition of *failure*.

Note

Base class version does nothing, returns zero.

Reimplemented from [std::basic_streambuf<_Elem, _Tr>](#).

Definition at line 374 of file `locale_conv.h`.

5.1034.4.35 `virtual int_type std::basic_streambuf<_Elem, _Tr>::uflow () [inline], [protected], [virtual], [inherited]`

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function does the same thing as `underflow()`, and in fact is required to call that function. It also returns the new character, like `underflow()` does. However, this function also moves the read position forward by one.

Definition at line 700 of file `streambuf`.

5.1034.4.36 `template<typename _Codecvt, typename _Elem = wchar_t, typename _Tr = char_traits<_Elem>>
_Wide_streambuf::int_type std::wbuffer_convert<_Codecvt, _Elem, _Tr>::underflow () [inline], [protected], [virtual]`

Fetches more data from the controlled sequence.

Returns

The first character from the *pending sequence*.

Informally, this function is called when the input buffer is exhausted (or does not exist, as buffering need not actually be done). If a buffer exists, it is *refilled*. In either case, the next available character is returned, or `traits::eof()` to indicate a null pending sequence.

For a formal definition of the pending sequence, see a good text such as Langer & Kreft, or [27.5.2.4.3]/7-14.

A functioning input `streambuf` can be created by overriding only this function (no buffer area will be used). For an example, see <https://gcc.gnu.org/onlinedocs/libstdc++/manual/streambufs.html>

Note

Base class version does nothing, returns `eof()`.

Reimplemented from `std::basic_streambuf<_Elem, _Tr>`.

Definition at line 388 of file `locale_conv.h`.

References `std::min()`.

5.1034.4.37 `virtual streamsize std::basic_streambuf<_Elem, _Tr>::xsgetn (char_type * __s, streamsize __n) [protected], [virtual], [inherited]`

Multiple character extraction.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to assign.

Returns

The number of characters assigned.

Fills `__s[0]` through `__s[__n-1]` with characters from the input sequence, as if by `sbumpc()`. Stops when either `__n` characters have been copied, or when `traits::eof()` would be copied.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

5.1034.4.38 `virtual streamsize std::basic_streambuf<_Elem, _Tr>::xspn(const char_type * __s, streamsize __n)`
`[protected], [virtual], [inherited]`

Multiple character insertion.

Parameters

<code>__s</code>	A buffer area.
<code>__n</code>	Maximum number of characters to write.

Returns

The number of characters written.

Writes `__s[0]` through `__s[__n-1]` to the output sequence, as if by `sputc()`. Stops when either `n` characters have been copied, or when `sputc()` would return `traits::eof()`.

It is expected that derived classes provide a more efficient implementation by overriding this definition.

5.1034.5 Member Data Documentation

5.1034.5.1 `locale std::basic_streambuf<_Elem, _Tr>::M_buf_locale` `[protected], [inherited]`

Current locale setting.

Definition at line 192 of file `streambuf`.

5.1034.5.2 `char_type* std::basic_streambuf<_Elem,_Tr>::_M_in_beg` [protected],[inherited]

Start of get area.

Definition at line 184 of file streambuf.

5.1034.5.3 `char_type* std::basic_streambuf<_Elem,_Tr>::_M_in_cur` [protected],[inherited]

Current read area.

Definition at line 185 of file streambuf.

5.1034.5.4 `char_type* std::basic_streambuf<_Elem,_Tr>::_M_in_end` [protected],[inherited]

End of get area.

Definition at line 186 of file streambuf.

5.1034.5.5 `char_type* std::basic_streambuf<_Elem,_Tr>::_M_out_beg` [protected],[inherited]

Start of put area.

Definition at line 187 of file streambuf.

5.1034.5.6 `char_type* std::basic_streambuf<_Elem,_Tr>::_M_out_cur` [protected],[inherited]

Current put area.

Definition at line 188 of file streambuf.

5.1034.5.7 `char_type* std::basic_streambuf<_Elem,_Tr>::_M_out_end` [protected],[inherited]

End of put area.

Definition at line 189 of file streambuf.

The documentation for this class was generated from the following file:

- [locale_conv.h](#)

5.1035 `std::weak_ptr<_Tp>` Class Template Reference

Inherits `std::__weak_ptr<_Tp,_Lp>`.

Public Types

- `typedef _Tp element_type`

Public Member Functions

- template<typename _Tp1, typename = _Convertible<_Tp1*>>
 weak_ptr (const [shared_ptr](#)<_Tp1> &__r) noexcept
- **weak_ptr** (const [weak_ptr](#) &) noexcept=default
- template<typename _Tp1, typename = _Convertible<_Tp1*>>
 weak_ptr (const [weak_ptr](#)<_Tp1> &__r) noexcept
- **weak_ptr** ([weak_ptr](#) &&) noexcept=default
- template<typename _Tp1, typename = _Convertible<_Tp1*>>
 weak_ptr ([weak_ptr](#)<_Tp1> &&__r) noexcept
- bool **expired** () const noexcept
- [shared_ptr](#)<_Tp> **lock** () const noexcept
- [weak_ptr](#) & **operator=** (const [weak_ptr](#) &__r) noexcept=default
- template<typename _Tp1>
 [weak_ptr](#) & **operator=** (const [weak_ptr](#)<_Tp1> &__r) noexcept
- template<typename _Tp1>
 [weak_ptr](#) & **operator=** (const [shared_ptr](#)<_Tp1> &__r) noexcept
- [weak_ptr](#) & **operator=** ([weak_ptr](#) &&__r) noexcept=default
- template<typename _Tp1>
 [weak_ptr](#) & **operator=** ([weak_ptr](#)<_Tp1> &&__r) noexcept
- template<typename _Tp1>
 bool **owner_before** (const __shared_ptr<_Tp1, _Lp> &__rhs) const
- template<typename _Tp1>
 bool **owner_before** (const __weak_ptr<_Tp1, _Lp> &__rhs) const
- void **reset** () noexcept
- void **swap** (__weak_ptr &__s) noexcept
- long **use_count** () const noexcept

5.1035.1 Detailed Description

```
template<typename _Tp>
class std::weak_ptr<_Tp>
```

A smart pointer with weak semantics.

With forwarding constructors and assignment operators.

Definition at line 470 of file bits/shared_ptr.h.

The documentation for this class was generated from the following file:

- [bits/shared_ptr.h](#)

5.1036 std::weak_ptr<_RealType> Class Template Reference

Classes

- struct [param_type](#)

Public Types

- typedef `_RealType` `result_type`

Public Member Functions

- **weibull_distribution** (`_RealType __a=_RealType(1), _RealType __b=_RealType(1)`)
- **weibull_distribution** (const `param_type` &__p)
- template<typename `_ForwardIterator` , typename `_UniformRandomNumberGenerator` >
void **generate** (`_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng`)
- template<typename `_ForwardIterator` , typename `_UniformRandomNumberGenerator` >
void **generate** (`_ForwardIterator __f, _ForwardIterator __t, _UniformRandomNumberGenerator &__urng, const param_type &__p`)
- template<typename `_UniformRandomNumberGenerator` >
void **generate** (`result_type *__f, result_type *__t, _UniformRandomNumberGenerator &__urng, const param_type &__p`)
- `_RealType a` () const
- `_RealType b` () const
- `result_type max` () const
- `result_type min` () const
- template<typename `_UniformRandomNumberGenerator` >
`result_type operator()` (`_UniformRandomNumberGenerator &__urng`)
- template<typename `_UniformRandomNumberGenerator` >
`result_type operator()` (`_UniformRandomNumberGenerator &__urng, const param_type &__p`)
- `param_type param` () const
- void `param` (const `param_type` &__param)
- void `reset` ()

Friends

- bool `operator==` (const `weibull_distribution` &__d1, const `weibull_distribution` &__d2)

5.1036.1 Detailed Description

```
template<typename _RealType = double>
class std::weibull_distribution< _RealType >
```

A `weibull_distribution` random number distribution.

The formula for the normal probability density function is:

$$p(x|\alpha, \beta) = \frac{\alpha}{\beta} \left(\frac{x}{\beta}\right)^{\alpha-1} \exp\left(-\left(\frac{x}{\beta}\right)^{\alpha}\right)$$

Definition at line 4683 of file `random.h`.

5.1036.2 Member Typedef Documentation

5.1036.2.1 `template<typename _RealType = double> typedef _RealType std::weibull_distribution<_RealType>::result_type`

The type of the range of the distribution.

Definition at line 4686 of file `random.h`.

5.1036.3 Member Function Documentation

5.1036.3.1 `template<typename _RealType = double> _RealType std::weibull_distribution<_RealType>::a () const`
[inline]

Return the a parameter of the distribution.

Definition at line 4741 of file `random.h`.

5.1036.3.2 `template<typename _RealType = double> _RealType std::weibull_distribution<_RealType>::b () const`
[inline]

Return the b parameter of the distribution.

Definition at line 4748 of file `random.h`.

5.1036.3.3 `template<typename _RealType = double> result_type std::weibull_distribution<_RealType>::max () const`
[inline]

Returns the least upper bound value of the distribution.

Definition at line 4777 of file `random.h`.

References `std::numeric_limits<_Tp>::max()`.

5.1036.3.4 `template<typename _RealType = double> result_type std::weibull_distribution<_RealType>::min () const`
[inline]

Returns the greatest lower bound value of the distribution.

Definition at line 4770 of file `random.h`.

5.1036.3.5 `template<typename _RealType = double> template<typename UniformRandomNumberGenerator> result_type`
`std::weibull_distribution<_RealType>::operator() (UniformRandomNumberGenerator & __urng)`
[inline]

Generating functions.

Definition at line 4785 of file `random.h`.

Referenced by `std::gamma_distribution<_RealType>::operator()()`.

5.1036.3.6 `template<typename _RealType = double> param_type std::weibull_distribution< _RealType >::param ()`
`const [inline]`

Returns the parameter set of the distribution.

Definition at line 4755 of file random.h.

Referenced by `std::operator>>()`.

5.1036.3.7 `template<typename _RealType = double> void std::weibull_distribution< _RealType >::param (const`
`param_type & __param) [inline]`

Sets the parameter set of the distribution.

Parameters

<code>__param</code>	The new parameter set of the distribution.
----------------------	--

Definition at line 4763 of file `random.h`.

5.1036.3.8 `template<typename _RealType = double> void std::weibull_distribution<_RealType>::reset () [inline]`

Resets the distribution state.

Definition at line 4734 of file `random.h`.

5.1036.4 Friends And Related Function Documentation

5.1036.4.1 `template<typename _RealType = double> bool operator==(const weibull_distribution<_RealType> & __d1, const weibull_distribution<_RealType> & __d2) [friend]`

Return true if two Weibull distributions have the same parameters.

Definition at line 4820 of file `random.h`.

The documentation for this class was generated from the following files:

- [random.h](#)
- [bits/random.tcc](#)

5.1037 `std::weibull_distribution<_RealType>::param_type` Struct Reference

Public Types

- typedef `weibull_distribution<_RealType>` **distribution_type**

Public Member Functions

- **param_type** (`_RealType __a=_RealType(1), _RealType __b=_RealType(1)`)
- `_RealType a` () const
- `_RealType b` () const

Friends

- bool **operator==** (const `param_type` &__p1, const `param_type` &__p2)

5.1037.1 Detailed Description

```
template<typename _RealType = double>
struct std::weibull_distribution< _RealType >::param_type
```

Parameter type.

Definition at line 4692 of file random.h.

The documentation for this struct was generated from the following file:

- [random.h](#)

5.1038 std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc > Class Template Reference

Public Types

- typedef [basic_string](#)< char, [char_traits](#)< char >, _Byte_alloc > **byte_string**
- typedef [wide_string::traits_type::int_type](#) **int_type**
- typedef [_Codecvt::state_type](#) **state_type**
- typedef [basic_string](#)< _Elem, [char_traits](#)< _Elem >, _Wide_alloc > **wide_string**

Public Member Functions

- [wstring_convert](#) (_Codecvt * __pcvt=new _Codecvt())
- [wstring_convert](#) (_Codecvt * __pcvt, state_type __state)
- [wstring_convert](#) (const [byte_string](#) & __byte_err, const [wide_string](#) & __wide_err=[wide_string](#)())
- [wstring_convert](#) (const [wstring_convert](#) &)=delete
- size_t [converted](#) () const noexcept
- [wstring_convert](#) & **operator=** (const [wstring_convert](#) &)=delete
- state_type [state](#) () const
- [wide_string from_bytes](#) (char __byte)
- [wide_string from_bytes](#) (const char * __ptr)
- [wide_string from_bytes](#) (const [byte_string](#) & __str)
- [wide_string from_bytes](#) (const char * __first, const char * __last)
- [byte_string to_bytes](#) (_Elem __wchar)
- [byte_string to_bytes](#) (const _Elem * __ptr)
- [byte_string to_bytes](#) (const [wide_string](#) & __wstr)
- [byte_string to_bytes](#) (const _Elem * __first, const _Elem * __last)

5.1038.1 Detailed Description

```
template<typename _Codecvt, typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>, typename _Byte_alloc =
allocator<char>>
class std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >
```

String conversions.

Definition at line 166 of file locale_conv.h.

5.1038.2 Constructor & Destructor Documentation

```
5.1038.2.1 template<typename _Codecvt, typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>, typename
_Byte_alloc = allocator<char>> std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc
>::wstring_convert ( _Codecvt * __pcvt = new _Codecvt () ) [inline], [explicit]
```

Default constructor.

Parameters

<code>__pcvt</code>	The facet to use for conversions.
---------------------	-----------------------------------

Takes ownership of `__pcvt` and will delete it in the destructor.

Definition at line 181 of file locale_conv.h.

Referenced by `std::wstring_convert<_Codecvt, _Elem, _Wide_alloc, _Byte_alloc>::wstring_convert()`.

```
5.1038.2.2 template<typename _Codecvt, typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>, typename
_Byte_alloc = allocator<char>> std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc
>::wstring_convert ( _Codecvt * __pcvt, state_type __state ) [inline]
```

Construct with an initial conversion state.

Parameters

<code>__pcvt</code>	The facet to use for conversions.
<code>__state</code>	Initial conversion state.

Takes ownership of `__pcvt` and will delete it in the destructor. The object's conversion state will persist between conversions.

Definition at line 195 of file locale_conv.h.

```
5.1038.2.3 template<typename _Codecvt, typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>, typename
    _Byte_alloc = allocator<char>> std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc
    >::wstring_convert( const byte_string & __byte_err, const wide_string & __wide_err = wide_string() )
    [inline], [explicit]
```

Construct with error strings.

Parameters

<code>__byte_err</code>	A string to return on failed conversions.
<code>__wide_err</code>	A wide string to return on failed conversions.

Definition at line 208 of file `locale_conv.h`.

References `std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::wstring_convert()`.

5.1038.3 Member Function Documentation

```
5.1038.3.1 template<typename _Codecvt, typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>, typename
    _Byte_alloc = allocator<char>> size_t std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc
    >::converted( ) const [inline], [noexcept]
```

The number of elements successfully converted in the last conversion.

Definition at line 298 of file `locale_conv.h`.

```
5.1038.3.2 template<typename _Codecvt, typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>, typename
    _Byte_alloc = allocator<char>> wide_string std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc
    >::from_bytes( char __byte ) [inline]
```

Convert from bytes.

Definition at line 227 of file `locale_conv.h`.

Referenced by `std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::from_bytes()`.

```
5.1038.3.3 template<typename _Codecvt, typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>, typename
    _Byte_alloc = allocator<char>> wide_string std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc
    >::from_bytes( const char * __ptr ) [inline]
```

Convert from bytes.

Definition at line 234 of file `locale_conv.h`.

References `std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::from_bytes()`.

```
5.1038.3.4 template<typename _Codecvt, typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>, typename
    _Byte_alloc = allocator<char>> wide_string std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc
    >::from_bytes ( const byte_string & __str ) [inline]
```

Convert from bytes.

Definition at line 238 of file locale_conv.h.

References std::basic_string< _CharT, _Traits, _Alloc >::data(), std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::from_bytes(), and std::basic_string< _CharT, _Traits, _Alloc >::size().

```
5.1038.3.5 template<typename _Codecvt, typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>, typename
    _Byte_alloc = allocator<char>> wide_string std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc
    >::from_bytes ( const char * __first, const char * __last ) [inline]
```

Convert from bytes.

Definition at line 245 of file locale_conv.h.

References std::basic_string< _CharT, _Traits, _Alloc >::get_allocator().

```
5.1038.3.6 template<typename _Codecvt, typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>, typename
    _Byte_alloc = allocator<char>> state_type std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc
    >::state ( ) const [inline]
```

The final conversion state of the last conversion.

Definition at line 301 of file locale_conv.h.

```
5.1038.3.7 template<typename _Codecvt, typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>, typename
    _Byte_alloc = allocator<char>> byte_string std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc
    >::to_bytes ( _Elem __wchar ) [inline]
```

Convert to bytes.

Definition at line 261 of file locale_conv.h.

Referenced by std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::to_bytes().

```
5.1038.3.8 template<typename _Codecvt, typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>, typename
    _Byte_alloc = allocator<char>> byte_string std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc
    >::to_bytes ( const _Elem * __ptr ) [inline]
```

Convert to bytes.

Definition at line 268 of file locale_conv.h.

References std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::to_bytes().

```
5.1038.3.9  template<typename _Codecvt, typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>, typename
             _Byte_alloc = allocator<char>>> byte_string std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc
             >::to_bytes ( const wide_string & __wstr )  [inline]
```

Convert to bytes.

Definition at line 274 of file locale_conv.h.

References `std::basic_string< _CharT, _Traits, _Alloc >::data()`, `std::basic_string< _CharT, _Traits, _Alloc >::size()`, and `std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >::to_bytes()`.

```
5.1038.3.10 template<typename _Codecvt, typename _Elem = wchar_t, typename _Wide_alloc = allocator<_Elem>, typename
             _Byte_alloc = allocator<char>>> byte_string std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc
             >::to_bytes ( const _Elem * __first, const _Elem * __last )  [inline]
```

Convert to bytes.

Definition at line 281 of file locale_conv.h.

References `std::basic_string< _CharT, _Traits, _Alloc >::get_allocator()`.

The documentation for this class was generated from the following file:

- [locale_conv.h](#)

6 File Documentation

6.1 algo.h File Reference

Classes

- struct `std::__parallel::CRandNumber< _MustBeInt >`

Namespaces

- `std`
- `std::__parallel`

Functions

- `template<typename _RAIter >`
`_RAIter std::__parallel::__adjacent_find_switch (_RAIter __begin, _RAIter __end, random_access_iterator_↵`
`_tag)`
- `template<typename _FIterator, typename _IteratorTag >`
`_FIterator std::__parallel::__adjacent_find_switch (_FIterator __begin, _FIterator __end, _IteratorTag)`
- `template<typename _FIterator, typename _BinaryPredicate, typename _IteratorTag >`
`_FIterator std::__parallel::__adjacent_find_switch (_FIterator __begin, _FIterator __end, _BinaryPredicate _↵`
`__pred, _IteratorTag)`
- `template<typename _RAIter, typename _BinaryPredicate >`
`_RAIter std::__parallel::__adjacent_find_switch (_RAIter __begin, _RAIter __end, _BinaryPredicate __pred,`
`random_access_iterator_tag)`
- `template<typename _RAIter, typename _Predicate >`
`iterator_traits< _RAIter >::difference_type std::__parallel::__count_if_switch (_RAIter __begin, _RAIter _↵`
`end, _Predicate __pred, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Predicate, typename _IteratorTag >`
`iterator_traits< _Iter >::difference_type std::__parallel::__count_if_switch (_Iter __begin, _Iter __end, _↵`
`Predicate __pred, _IteratorTag)`
- `template<typename _RAIter, typename _Tp >`
`iterator_traits< _RAIter >::difference_type std::__parallel::__count_switch (_RAIter __begin, _RAIter __end,`
`const _Tp &__value, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Tp, typename _IteratorTag >`
`iterator_traits< _Iter >::difference_type std::__parallel::__count_switch (_Iter __begin, _Iter __end, const`
`_Tp &__value, _IteratorTag)`
- `template<typename _Iter, typename _FIterator, typename _IteratorTag1, typename _IteratorTag2 >`
`_Iter std::__parallel::__find_first_of_switch (_Iter __begin1, _Iter __end1, _FIterator __begin2, _FIterator`
`__end2, _IteratorTag1, _IteratorTag2)`
- `template<typename _RAIter, typename _FIterator, typename _BinaryPredicate, typename _IteratorTag >`
`_RAIter std::__parallel::__find_first_of_switch (_RAIter __begin1, _RAIter __end1, _FIterator __begin2, _F↵`
`Iterator __end2, _BinaryPredicate __comp, random_access_iterator_tag, _IteratorTag)`
- `template<typename _Iter, typename _FIterator, typename _BinaryPredicate, typename _IteratorTag1, typename _IteratorTag2 >`
`_Iter std::__parallel::__find_first_of_switch (_Iter __begin1, _Iter __end1, _FIterator __begin2, _FIterator`
`__end2, _BinaryPredicate __comp, _IteratorTag1, _IteratorTag2)`
- `template<typename _Iter, typename _Predicate, typename _IteratorTag >`
`_Iter std::__parallel::__find_if_switch (_Iter __begin, _Iter __end, _Predicate __pred, _IteratorTag)`
- `template<typename _RAIter, typename _Predicate >`
`_RAIter std::__parallel::__find_if_switch (_RAIter __begin, _RAIter __end, _Predicate __pred, random_↵`
`access_iterator_tag)`
- `template<typename _Iter, typename _Tp, typename _IteratorTag >`
`_Iter std::__parallel::__find_switch (_Iter __begin, _Iter __end, const _Tp &__val, _IteratorTag)`
- `template<typename _RAIter, typename _Tp >`
`_RAIter std::__parallel::__find_switch (_RAIter __begin, _RAIter __end, const _Tp &__val, random_access_↵`
`iterator_tag)`
- `template<typename _Iter, typename _Function, typename _IteratorTag >`
`_Function std::__parallel::__for_each_switch (_Iter __begin, _Iter __end, _Function __f, _IteratorTag)`
- `template<typename _RAIter, typename _Function >`
`_Function std::__parallel::__for_each_switch (_RAIter __begin, _RAIter __end, _Function __f, random_↵`
`access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _OutputIterator, typename _Size, typename _Generator, typename _IteratorTag >`
`_OutputIterator std::__parallel::__generate_n_switch (_OutputIterator __begin, _Size __n, _Generator __gen,`
`_IteratorTag)`

- `template<typename _RAIter, typename _Size, typename _Generator >`
`_RAIter std::parallel::generate_n_switch (_RAIter __begin, _Size __n, _Generator __gen, random_↵`
`access_iterator_tag, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filterator, typename _Generator, typename _IteratorTag >`
`void std::parallel::generate_switch (_Filterator __begin, _Filterator __end, _Generator __gen, _IteratorTag)`
- `template<typename _RAIter, typename _Generator >`
`void std::parallel::generate_switch (_RAIter __begin, _RAIter __end, _Generator __gen, random_↵`
`access_iterator_tag, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filterator, typename _Compare, typename _IteratorTag >`
`_Filterator std::parallel::max_element_switch (_Filterator __begin, _Filterator __end, _Compare __comp,`
`_IteratorTag)`
- `template<typename _RAIter, typename _Compare >`
`_RAIter std::parallel::max_element_switch (_RAIter __begin, _RAIter __end, _Compare __comp,`
`random_access_iterator_tag, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Compare, typename _IteratorTag1, typename _↵`
`IteratorTag2, typename _IteratorTag3 >`
`_OutputIterator std::parallel::merge_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 ↵`
`__end2, _OutputIterator __result, _Compare __comp, _IteratorTag1, _IteratorTag2, _IteratorTag3)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::parallel::merge_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 ↵`
`__end2, _OutputIterator __result, _Compare __comp, random_access_iterator_tag, random_access_iterator_tag,`
`random_access_iterator_tag)`
- `template<typename _Filterator, typename _Compare, typename _IteratorTag >`
`_Filterator std::parallel::min_element_switch (_Filterator __begin, _Filterator __end, _Compare __comp,`
`_IteratorTag)`
- `template<typename _RAIter, typename _Compare >`
`_RAIter std::parallel::min_element_switch (_RAIter __begin, _RAIter __end, _Compare __comp,`
`random_access_iterator_tag, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filterator, typename _Predicate, typename _IteratorTag >`
`_Filterator std::parallel::partition_switch (_Filterator __begin, _Filterator __end, _Predicate __pred, ↵`
`IteratorTag)`
- `template<typename _RAIter, typename _Predicate >`
`_RAIter std::parallel::partition_switch (_RAIter __begin, _RAIter __end, _Predicate __pred, random_↵`
`access_iterator_tag)`
- `template<typename _Filterator, typename _Predicate, typename _Tp, typename _IteratorTag >`
`void std::parallel::replace_if_switch (_Filterator __begin, _Filterator __end, _Predicate __pred, const _Tp`
`& __new_value, _IteratorTag)`
- `template<typename _RAIter, typename _Predicate, typename _Tp >`
`void std::parallel::replace_if_switch (_RAIter __begin, _RAIter __end, _Predicate __pred, const _Tp &↵`
`__new_value, random_access_iterator_tag, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filterator, typename _Tp, typename _IteratorTag >`
`void std::parallel::replace_switch (_Filterator __begin, _Filterator __end, const _Tp & __old_value, const`
`_Tp & __new_value, _IteratorTag)`
- `template<typename _RAIter, typename _Tp >`
`void std::parallel::replace_switch (_RAIter __begin, _RAIter __end, const _Tp & __old_value, const _Tp`
`& __new_value, random_access_iterator_tag, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _RAIter, typename _Integer, typename _Tp, typename _BinaryPredicate >`
`_RAIter std::parallel::search_n_switch (_RAIter __begin, _RAIter __end, _Integer __count, const _Tp &↵`
`__val, _BinaryPredicate __binary_pred, random_access_iterator_tag)`
- `template<typename _Filterator, typename _Integer, typename _Tp, typename _BinaryPredicate, typename _IteratorTag >`
`_Filterator std::parallel::search_n_switch (_Filterator __begin, _Filterator __end, _Integer __count, const`
`_Tp & __val, _BinaryPredicate __binary_pred, _IteratorTag)`

- `template<typename _RAIter1, typename _RAIter2 >`
`_RAIter1 std::parallel::search_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2`
`__end2, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _FIterator1, typename _FIterator2, typename _IteratorTag1, typename _IteratorTag2 >`
`_FIterator1 std::parallel::search_switch (_FIterator1 __begin1, _FIterator1 __end1, _FIterator2 __begin2,`
`_FIterator2 __end2, _IteratorTag1, _IteratorTag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _BinaryPredicate >`
`_RAIter1 std::parallel::search_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2`
`__end2, _BinaryPredicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _FIterator1, typename _FIterator2, typename _BinaryPredicate, typename _IteratorTag1, typename _IteratorTag2 >`
`_FIterator1 std::parallel::search_switch (_FIterator1 __begin1, _FIterator1 __end1, _FIterator2 __begin2,`
`_FIterator2 __end2, _BinaryPredicate __pred, _IteratorTag1, _IteratorTag2)`
- `template<typename _IIter1, typename _IIter2, typename _Predicate, typename _OutputIterator, typename _IteratorTag1, typename _IteratorTag2, typename _IteratorTag3 >`
`_OutputIterator std::parallel::set_difference_switch (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2,`
`_IIter2 __end2, _OutputIterator __result, _Predicate __pred, _IteratorTag1, _IteratorTag2, _IteratorTag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _Output_RAlter, typename _Predicate >`
`_Output_RAlter std::parallel::set_difference_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2,`
`_RAIter2 __end2, _Output_RAlter __result, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _IIter1, typename _IIter2, typename _Predicate, typename _OutputIterator, typename _IteratorTag1, typename _IteratorTag2, typename _IteratorTag3 >`
`_OutputIterator std::parallel::set_intersection_switch (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2,`
`_IIter2 __end2, _OutputIterator __result, _Predicate __pred, _IteratorTag1, _IteratorTag2, _IteratorTag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _Output_RAlter, typename _Predicate >`
`_Output_RAlter std::parallel::set_intersection_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2,`
`_RAIter2 __end2, _Output_RAlter __result, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _IIter1, typename _IIter2, typename _Predicate, typename _OutputIterator, typename _IteratorTag1, typename _IteratorTag2, typename _IteratorTag3 >`
`_OutputIterator std::parallel::set_symmetric_difference_switch (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2,`
`_IIter2 __end2, _OutputIterator __result, _Predicate __pred, _IteratorTag1, _IteratorTag2, _IteratorTag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _Output_RAlter, typename _Predicate >`
`_Output_RAlter std::parallel::set_symmetric_difference_switch (_RAIter1 __begin1, _RAIter1 __end1,`
`_RAIter2 __begin2, _RAIter2 __end2, _Output_RAlter __result, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _IIter1, typename _IIter2, typename _Predicate, typename _OutputIterator, typename _IteratorTag1, typename _IteratorTag2, typename _IteratorTag3 >`
`_OutputIterator std::parallel::set_union_switch (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2`
`__end2, _OutputIterator __result, _Predicate __pred, _IteratorTag1, _IteratorTag2, _IteratorTag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _Output_RAlter, typename _Predicate >`
`_Output_RAlter std::parallel::set_union_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2,`
`_RAIter2 __end2, _Output_RAlter __result, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _RAIter1, typename _RAIter2, typename _UnaryOperation >`
`_RAIter2 std::parallel::transform1_switch (_RAIter1 __begin, _RAIter1 __end, _RAIter2 __result, _UnaryOperation`
`__unary_op, random_access_iterator_tag, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _RAIter1, typename _RAIter2, typename _UnaryOperation, typename _IteratorTag1, typename _IteratorTag2 >`
`_RAIter2 std::parallel::transform1_switch (_RAIter1 __begin, _RAIter1 __end, _RAIter2 __result, _UnaryOperation`
`__unary_op, _IteratorTag1, _IteratorTag2)`

- `template<typename _RAIter1, typename _RAIter2, typename _RAIter3, typename _BinaryOperation >`
`_RAIter3 std::parallel::transform2_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, ↵`
`_RAIter3 __result, _BinaryOperation __binary_op, random_access_iterator_tag, random_access_iterator_tag,`
`random_access_iterator_tag, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator, typename _BinaryOperation, typename _Tag1, typename _Tag2`
`, typename _Tag3 >`
`_OutputIterator std::parallel::transform2_switch (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, ↵`
`_OutputIterator __result, _BinaryOperation __binary_op, _Tag1, _Tag2, _Tag3)`
- `template<typename _IIter, typename _OutputIterator, typename _Predicate, typename _IteratorTag1, typename _IteratorTag2 >`
`_OutputIterator std::parallel::unique_copy_switch (_IIter __begin, _IIter __last, _OutputIterator __out, ↵`
`_Predicate __pred, _IteratorTag1, _IteratorTag2)`
- `template<typename _RAIter, typename RandomAccessOutputIterator, typename _Predicate >`
`RandomAccessOutputIterator std::parallel::unique_copy_switch (_RAIter __begin, _RAIter __last,`
`RandomAccessOutputIterator __out, _Predicate __pred, random_access_iterator_tag, random_access_↵`
`iterator_tag)`
- `template<typename _FIterator >`
`_FIterator std::parallel::adjacent_find (_FIterator __begin, _FIterator __end, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator, typename _BinaryPredicate >`
`_FIterator std::parallel::adjacent_find (_FIterator __begin, _FIterator __end, _BinaryPredicate __binary_↵`
`pred, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator >`
`_FIterator std::parallel::adjacent_find (_FIterator __begin, _FIterator __end)`
- `template<typename _FIterator, typename _BinaryPredicate >`
`_FIterator std::parallel::adjacent_find (_FIterator __begin, _FIterator __end, _BinaryPredicate __pred)`
- `template<typename _IIter, typename _Tp >`
`iterator_traits< _IIter >::difference_type std::parallel::count (_IIter __begin, _IIter __end, const _Tp &↵`
`value, __gnu_parallel::sequential_tag)`
- `template<typename _IIter, typename _Tp >`
`iterator_traits< _IIter >::difference_type std::parallel::count (_IIter __begin, _IIter __end, const _Tp &↵`
`value, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _IIter, typename _Tp >`
`iterator_traits< _IIter >::difference_type std::parallel::count (_IIter __begin, _IIter __end, const _Tp &↵`
`value)`
- `template<typename _IIter, typename _Predicate >`
`iterator_traits< _IIter >::difference_type std::parallel::count_if (_IIter __begin, _IIter __end, _Predicate __↵`
`pred, __gnu_parallel::sequential_tag)`
- `template<typename _IIter, typename _Predicate >`
`iterator_traits< _IIter >::difference_type std::parallel::count_if (_IIter __begin, _IIter __end, _Predicate __↵`
`pred, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _IIter, typename _Predicate >`
`iterator_traits< _IIter >::difference_type std::parallel::count_if (_IIter __begin, _IIter __end, _Predicate __↵`
`pred)`
- `template<typename _IIter, typename _Tp >`
`_IIter std::parallel::find (_IIter __begin, _IIter __end, const _Tp &__val, __gnu_parallel::sequential_tag)`
- `template<typename _IIter, typename _Tp >`
`_IIter std::parallel::find (_IIter __begin, _IIter __end, const _Tp &__val)`
- `template<typename _IIter, typename _FIterator >`
`_IIter std::parallel::find_first_of (_IIter __begin1, _IIter __end1, _FIterator __begin2, _FIterator __end2, ↵`
`__gnu_parallel::sequential_tag)`
- `template<typename _IIter, typename _FIterator, typename _BinaryPredicate >`
`_IIter std::parallel::find_first_of (_IIter __begin1, _IIter __end1, _FIterator __begin2, _FIterator __end2, ↵`
`_BinaryPredicate __comp, __gnu_parallel::sequential_tag)`

- `template<typename _Iter, typename _Filterator, typename _BinaryPredicate >`
`_Iter std::parallel::find_first_of (_Iter __begin1, _Iter __end1, _Filterator __begin2, _Filterator __end2, _↵`
`BinaryPredicate __comp)`
- `template<typename _Iter, typename _Filterator >`
`_Iter std::parallel::find_first_of (_Iter __begin1, _Iter __end1, _Filterator __begin2, _Filterator __end2)`
- `template<typename _Iter, typename _Predicate >`
`_Iter std::parallel::find_if (_Iter __begin, _Iter __end, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Predicate >`
`_Iter std::parallel::find_if (_Iter __begin, _Iter __end, _Predicate __pred)`
- `template<typename _Iter, typename _Function >`
`_Function std::parallel::for_each (_Iter __begin, _Iter __end, _Function __f, __gnu_parallel::sequential_tag)`
- `template<typename _Iterator, typename _Function >`
`_Function std::parallel::for_each (_Iterator __begin, _Iterator __end, _Function __f, __gnu_parallel::↵`
`Parallelism __parallelism_tag)`
- `template<typename _Iterator, typename _Function >`
`_Function std::parallel::for_each (_Iterator __begin, _Iterator __end, _Function __f)`
- `template<typename _Filterator, typename _Generator >`
`void std::parallel::generate (_Filterator __begin, _Filterator __end, _Generator __gen, __gnu_parallel::↵`
`sequential_tag)`
- `template<typename _Filterator, typename _Generator >`
`void std::parallel::generate (_Filterator __begin, _Filterator __end, _Generator __gen, __gnu_parallel::↵`
`Parallelism __parallelism_tag)`
- `template<typename _Filterator, typename _Generator >`
`void std::parallel::generate (_Filterator __begin, _Filterator __end, _Generator __gen)`
- `template<typename _OutputIterator, typename _Size, typename _Generator >`
`_OutputIterator std::parallel::generate_n (_OutputIterator __begin, _Size __n, _Generator __gen, __gnu↵`
`parallel::sequential_tag)`
- `template<typename _OutputIterator, typename _Size, typename _Generator >`
`_OutputIterator std::parallel::generate_n (_OutputIterator __begin, _Size __n, _Generator __gen, __gnu↵`
`parallel::Parallelism __parallelism_tag)`
- `template<typename _OutputIterator, typename _Size, typename _Generator >`
`_OutputIterator std::parallel::generate_n (_OutputIterator __begin, _Size __n, _Generator __gen)`
- `template<typename _Filterator >`
`_Filterator std::parallel::max_element (_Filterator __begin, _Filterator __end, __gnu_parallel::sequential_tag)`
- `template<typename _Filterator, typename _Compare >`
`_Filterator std::parallel::max_element (_Filterator __begin, _Filterator __end, _Compare __comp, __gnu↵`
`parallel::sequential_tag)`
- `template<typename _Filterator >`
`_Filterator std::parallel::max_element (_Filterator __begin, _Filterator __end, __gnu_parallel::Parallelism ↵`
`__parallelism_tag)`
- `template<typename _Filterator >`
`_Filterator std::parallel::max_element (_Filterator __begin, _Filterator __end)`
- `template<typename _Filterator, typename _Compare >`
`_Filterator std::parallel::max_element (_Filterator __begin, _Filterator __end, _Compare __comp, __gnu↵`
`parallel::Parallelism __parallelism_tag)`
- `template<typename _Filterator, typename _Compare >`
`_Filterator std::parallel::max_element (_Filterator __begin, _Filterator __end, _Compare __comp)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator std::parallel::merge (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _↵`
`OutputIterator __result, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::parallel::merge (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _↵`
`OutputIterator __result, _Compare __comp, __gnu_parallel::sequential_tag)`

- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::parallel::merge (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, ↵`
`_OutputIterator __result, _Compare __comp)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator std::parallel::merge (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, ↵`
`_OutputIterator __result)`
- `template<typename _Filterator >`
`_Filterator std::parallel::min_element (_Filterator __begin, _Filterator __end, __gnu_parallel::sequential_tag)`
- `template<typename _Filterator, typename _Compare >`
`_Filterator std::parallel::min_element (_Filterator __begin, _Filterator __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _Filterator >`
`_Filterator std::parallel::min_element (_Filterator __begin, _Filterator __end, __gnu_parallel::Parallelism ↵`
`_parallelism_tag)`
- `template<typename _Filterator >`
`_Filterator std::parallel::min_element (_Filterator __begin, _Filterator __end)`
- `template<typename _Filterator, typename _Compare >`
`_Filterator std::parallel::min_element (_Filterator __begin, _Filterator __end, _Compare __comp, __gnu_parallel::Parallelism ↵`
`_parallelism_tag)`
- `template<typename _Filterator, typename _Compare >`
`_Filterator std::parallel::min_element (_Filterator __begin, _Filterator __end, _Compare __comp)`
- `template<typename _RAlter >`
`void std::parallel::nth_element (_RAlter __begin, _RAlter __nth, _RAlter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAlter, typename _Compare >`
`void std::parallel::nth_element (_RAlter __begin, _RAlter __nth, _RAlter __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _RAlter, typename _Compare >`
`void std::parallel::nth_element (_RAlter __begin, _RAlter __nth, _RAlter __end, _Compare __comp)`
- `template<typename _RAlter >`
`void std::parallel::nth_element (_RAlter __begin, _RAlter __nth, _RAlter __end)`
- `template<typename _RAlter, typename _Compare >`
`void std::parallel::partial_sort (_RAlter __begin, _RAlter __middle, _RAlter __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _RAlter >`
`void std::parallel::partial_sort (_RAlter __begin, _RAlter __middle, _RAlter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAlter, typename _Compare >`
`void std::parallel::partial_sort (_RAlter __begin, _RAlter __middle, _RAlter __end, _Compare __comp)`
- `template<typename _RAlter >`
`void std::parallel::partial_sort (_RAlter __begin, _RAlter __middle, _RAlter __end)`
- `template<typename _Filterator, typename _Predicate >`
`_Filterator std::parallel::partition (_Filterator __begin, _Filterator __end, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Filterator, typename _Predicate >`
`_Filterator std::parallel::partition (_Filterator __begin, _Filterator __end, _Predicate __pred)`
- `template<typename _RAlter >`
`void std::parallel::random_shuffle (_RAlter __begin, _RAlter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAlter, typename _RandomNumberGenerator >`
`void std::parallel::random_shuffle (_RAlter __begin, _RAlter __end, _RandomNumberGenerator &__rand, __gnu_parallel::sequential_tag)`
- `template<typename _RAlter >`
`void std::parallel::random_shuffle (_RAlter __begin, _RAlter __end)`

- `template<typename _RAIter, typename _RandomNumberGenerator >`
`void std::__parallel::random_shuffle (_RAIter __begin, _RAIter __end, _RandomNumberGenerator &&__rand)`
- `template<typename _FIterator, typename _Tp >`
`void std::__parallel::replace (_FIterator __begin, _FIterator __end, const _Tp &__old_value, const _Tp &__new_value, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator, typename _Tp >`
`void std::__parallel::replace (_FIterator __begin, _FIterator __end, const _Tp &__old_value, const _Tp &__new_value, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _FIterator, typename _Tp >`
`void std::__parallel::replace (_FIterator __begin, _FIterator __end, const _Tp &__old_value, const _Tp &__new_value)`
- `template<typename _FIterator, typename _Predicate, typename _Tp >`
`void std::__parallel::replace_if (_FIterator __begin, _FIterator __end, _Predicate __pred, const _Tp &__new_value, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator, typename _Predicate, typename _Tp >`
`void std::__parallel::replace_if (_FIterator __begin, _FIterator __end, _Predicate __pred, const _Tp &__new_value, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _FIterator, typename _Predicate, typename _Tp >`
`void std::__parallel::replace_if (_FIterator __begin, _FIterator __end, _Predicate __pred, const _Tp &__new_value)`
- `template<typename _FIterator1, typename _FIterator2 >`
`_FIterator1 std::__parallel::search (_FIterator1 __begin1, _FIterator1 __end1, _FIterator2 __begin2, _FIterator2 __end2, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator1, typename _FIterator2 >`
`_FIterator1 std::__parallel::search (_FIterator1 __begin1, _FIterator1 __end1, _FIterator2 __begin2, _FIterator2 __end2)`
- `template<typename _FIterator1, typename _FIterator2, typename _BinaryPredicate >`
`_FIterator1 std::__parallel::search (_FIterator1 __begin1, _FIterator1 __end1, _FIterator2 __begin2, _FIterator2 __end2, _BinaryPredicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator1, typename _FIterator2, typename _BinaryPredicate >`
`_FIterator1 std::__parallel::search (_FIterator1 __begin1, _FIterator1 __end1, _FIterator2 __begin2, _FIterator2 __end2, _BinaryPredicate __pred)`
- `template<typename _FIterator, typename _Integer, typename _Tp >`
`_FIterator std::__parallel::search_n (_FIterator __begin, _FIterator __end, _Integer __count, const _Tp &__val, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator, typename _Integer, typename _Tp, typename _BinaryPredicate >`
`_FIterator std::__parallel::search_n (_FIterator __begin, _FIterator __end, _Integer __count, const _Tp &__val, _BinaryPredicate __binary_pred, __gnu_parallel::sequential_tag)`
- `template<typename _FIterator, typename _Integer, typename _Tp >`
`_FIterator std::__parallel::search_n (_FIterator __begin, _FIterator __end, _Integer __count, const _Tp &__val)`
- `template<typename _FIterator, typename _Integer, typename _Tp, typename _BinaryPredicate >`
`_FIterator std::__parallel::search_n (_FIterator __begin, _FIterator __end, _Integer __count, const _Tp &__val, _BinaryPredicate __binary_pred)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator >`
`_OutputIterator std::__parallel::set_difference (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 __end2, _OutputIterator __out, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::__parallel::set_difference (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 __end2, _OutputIterator __out, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _OutputIterator >`
`_OutputIterator std::__parallel::set_difference (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _IIter2 __end2, _OutputIterator __out)`

- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::parallel::set_difference (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, _Predicate __pred)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator std::parallel::set_intersection (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::parallel::set_intersection (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, _Predicate __pred, gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator std::parallel::set_intersection (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::parallel::set_intersection (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, _Predicate __pred)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator std::parallel::set_symmetric_difference (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::parallel::set_symmetric_difference (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, _Predicate __pred, gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator std::parallel::set_symmetric_difference (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::parallel::set_symmetric_difference (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, _Predicate __pred)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator std::parallel::set_union (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::parallel::set_union (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, _Predicate __pred, gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator >`
`_OutputIterator std::parallel::set_union (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::parallel::set_union (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _OutputIterator __out, _Predicate __pred)`
- `template<typename _RAIter >`
`void std::parallel::sort (_RAIter __begin, _RAIter __end, gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare >`
`void std::parallel::sort (_RAIter __begin, _RAIter __end, _Compare __comp, gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare, typename _Parallelism >`
`void std::parallel::sort (_RAIter __begin, _RAIter __end, _Compare __comp, _Parallelism __parallelism)`
- `template<typename _RAIter >`
`void std::parallel::sort (_RAIter __begin, _RAIter __end)`
- `template<typename _RAIter >`
`void std::parallel::sort (_RAIter __begin, _RAIter __end, gnu_parallel::default_parallel_tag __parallelism)`
- `template<typename _RAIter >`
`void std::parallel::sort (_RAIter __begin, _RAIter __end, gnu_parallel::parallel_tag __parallelism)`
- `template<typename _RAIter >`
`void std::parallel::sort (_RAIter __begin, _RAIter __end, gnu_parallel::multiway_mergesort_tag __parallelism)`

- `template<typename _RAIter >`
`void std::__parallel::sort (_RAIter __begin, _RAIter __end, __gnu_parallel::multiway_mergesort_sampling_tag __parallelism)`
- `template<typename _RAIter >`
`void std::__parallel::sort (_RAIter __begin, _RAIter __end, __gnu_parallel::multiway_mergesort_exact_tag __parallelism)`
- `template<typename _RAIter >`
`void std::__parallel::sort (_RAIter __begin, _RAIter __end, __gnu_parallel::quicksort_tag __parallelism)`
- `template<typename _RAIter >`
`void std::__parallel::sort (_RAIter __begin, _RAIter __end, __gnu_parallel::balanced_quicksort_tag __parallelism)`
- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::sort (_RAIter __begin, _RAIter __end, _Compare __comp)`
- `template<typename _RAIter >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare, typename _Parallelism >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, _Compare __comp, _Parallelism __parallelism)`
- `template<typename _RAIter >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end)`
- `template<typename _RAIter >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::default_parallel_tag __parallelism)`
- `template<typename _RAIter >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::parallel_tag __parallelism)`
- `template<typename _RAIter >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::multiway_mergesort_tag __parallelism)`
- `template<typename _RAIter >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::quicksort_tag __parallelism)`
- `template<typename _RAIter >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::balanced_quicksort_tag __parallelism)`
- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, _Compare __comp)`
- `template<typename _Iter, typename _OutputIterator, typename _UnaryOperation >`
`_OutputIterator std::__parallel::transform (_Iter __begin, _Iter __end, _OutputIterator __result, _UnaryOperation __unary_op, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _UnaryOperation >`
`_OutputIterator std::__parallel::transform (_Iter __begin, _Iter __end, _OutputIterator __result, _UnaryOperation __unary_op, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _UnaryOperation >`
`_OutputIterator std::__parallel::transform (_Iter __begin, _Iter __end, _OutputIterator __result, _UnaryOperation __unary_op)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::__parallel::transform (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _OutputIterator __result, _BinaryOperation __binary_op, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::__parallel::transform (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _OutputIterator __result, _BinaryOperation __binary_op, __gnu_parallel::Parallelism __parallelism_tag)`

- `template<typename _Iter1, typename _Iter2, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::parallel::transform (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _OutputIterator`
`__result, _BinaryOperation __binary_op)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator std::parallel::unique_copy (_Iter __begin1, _Iter __end1, _OutputIterator __out, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::parallel::unique_copy (_Iter __begin1, _Iter __end1, _OutputIterator __out, _Predicate`
`__pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator std::parallel::unique_copy (_Iter __begin1, _Iter __end1, _OutputIterator __out)`
- `template<typename _Iter, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::parallel::unique_copy (_Iter __begin1, _Iter __end1, _OutputIterator __out, _Predicate`
`__pred)`

6.1.1 Detailed Description

Parallel STL function calls corresponding to the `stl_algo.h` header.

The functions defined here mainly do case switches and call the actual parallelized versions in other files. Inlining policy: Functions that basically only contain one function call, are declared inline. This file is a GNU parallel extension to the Standard C++ Library.

6.2 `algbase.h` File Reference

Namespaces

- `std`
- `std::parallel`

Functions

- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IteratorTag1, typename _IteratorTag2 >`
`bool std::parallel::equal_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, __gnu_parallel::equal_switch_tag, _Predicate __pred, _IteratorTag1, _IteratorTag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _Predicate >`
`bool std::parallel::equal_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, __gnu_parallel::equal_switch_tag, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IteratorTag1, typename _IteratorTag2 >`
`bool std::parallel::lexicographical_compare_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _Predicate __pred, _IteratorTag1, _IteratorTag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _Predicate >`
`bool std::parallel::lexicographical_compare_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IteratorTag1, typename _IteratorTag2 >`
`pair<_Iter1, _Iter2> std::parallel::mismatch_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Predicate __pred, _IteratorTag1, _IteratorTag2)`

- `template<typename _RAIter1, typename _RAIter2, typename _Predicate >`
`pair< _RAIter1, _RAIter2 > std::parallel::mismatch_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IteratorTag1, typename _IteratorTag2 >`
`pair< _Iter1, _Iter2 > std::parallel::mismatch_switch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _Predicate __pred, _IteratorTag1, _IteratorTag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _Predicate >`
`pair< _RAIter1, _RAIter2 > std::parallel::mismatch_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2 __end2, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`bool std::parallel::equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`bool std::parallel::equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`bool std::parallel::equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`bool std::parallel::equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Predicate __pred)`
- `template<typename _Iter1, typename _Iter2 >`
`bool std::parallel::equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _BinaryPredicate >`
`bool std::parallel::equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _BinaryPredicate __binary_pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`bool std::parallel::equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2)`
- `template<typename _Iter1, typename _Iter2, typename _BinaryPredicate >`
`bool std::parallel::equal (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _BinaryPredicate __binary_pred)`
- `template<typename _Iter1, typename _Iter2 >`
`bool std::parallel::lexicographical_compare (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`bool std::parallel::lexicographical_compare (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`bool std::parallel::lexicographical_compare (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`bool std::parallel::lexicographical_compare (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _Predicate __pred)`
- `template<typename _Iter1, typename _Iter2 >`
`pair< _Iter1, _Iter2 > std::parallel::mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`pair< _Iter1, _Iter2 > std::parallel::mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`pair< _Iter1, _Iter2 > std::parallel::mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`pair< _Iter1, _Iter2 > std::parallel::mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Predicate __pred)`

- `template<typename _InputIterator1, typename _InputIterator2 >`
`pair< _InputIterator1, _InputIterator2 > std::__parallel::mismatch (_InputIterator1 __first1, _InputIterator1 ↵`
`__last1, _InputIterator2 __first2, _InputIterator2 __last2, __gnu_parallel::sequential_tag)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`pair< _InputIterator1, _InputIterator2 > std::__parallel::mismatch (_InputIterator1 __first1, _InputIterator1 ↵`
`__last1, _InputIterator2 __first2, _InputIterator2 __last2, _BinaryPredicate __binary_pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`pair< _Iter1, _Iter2 > std::__parallel::mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 ↵`
`__end2)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`pair< _InputIterator1, _InputIterator2 > std::__parallel::mismatch (_InputIterator1 __begin1, _InputIterator1 ↵`
`__end1, _InputIterator2 __begin2, _InputIterator2 __end2, _BinaryPredicate __binary_pred)`

6.2.1 Detailed Description

Parallel STL function calls corresponding to the `stl_algobase.h` header. The functions defined here mainly do case switches and call the actual parallelized versions in other files. Inlining policy: Functions that basically only contain one function call, are declared inline. This file is a GNU parallel extension to the Standard C++ Library.

6.3 algorithm File Reference

Macros

- `#define _GLIBCXX_ALGORITHM`

6.3.1 Detailed Description

This is a Standard C++ Library header.

6.4 algorithm File Reference

Namespaces

- [__gnu_cxx](#)

Macros

- `#define _EXT_ALGORITHM`

Functions

- `template<typename _InputIterator, typename _Size, typename _OutputIterator >`
`pair< _InputIterator, _OutputIterator > __gnu_cxx::__copy_n (_InputIterator __first, _Size __count, _OutputIterator __result, input_iterator_tag)`
- `template<typename _RAIterator, typename _Size, typename _OutputIterator >`
`pair< _RAIterator, _OutputIterator > __gnu_cxx::__copy_n (_RAIterator __first, _Size __count, _OutputIterator __result, random_access_iterator_tag)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`int __gnu_cxx::__lexicographical_compare_3way (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2)`
- `int __gnu_cxx::__lexicographical_compare_3way (const unsigned char * __first1, const unsigned char * __last1, const unsigned char * __first2, const unsigned char * __last2)`
- `int __gnu_cxx::__lexicographical_compare_3way (const char * __first1, const char * __last1, const char * __first2, const char * __last2)`
- `template<typename _Tp >`
`const _Tp & __gnu_cxx::__median (const _Tp & __a, const _Tp & __b, const _Tp & __c)`
- `template<typename _Tp, typename _Compare >`
`const _Tp & __gnu_cxx::__median (const _Tp & __a, const _Tp & __b, const _Tp & __c, _Compare __comp)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _Distance >`
`_RandomAccessIterator __gnu_cxx::__random_sample (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out, const _Distance __n)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _RandomNumberGenerator, typename _Distance >`
`_RandomAccessIterator __gnu_cxx::__random_sample (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out, _RandomNumberGenerator & __rand, const _Distance __n)`
- `template<typename _InputIterator, typename _Size, typename _OutputIterator >`
`pair< _InputIterator, _OutputIterator > __gnu_cxx::copy_n (_InputIterator __first, _Size __count, _OutputIterator __result)`
- `template<typename _InputIterator, typename _Tp, typename _Size >`
`void __gnu_cxx::count (_InputIterator __first, _InputIterator __last, const _Tp & __value, _Size & __n)`
- `template<typename _InputIterator, typename _Predicate, typename _Size >`
`void __gnu_cxx::count_if (_InputIterator __first, _InputIterator __last, _Predicate __pred, _Size & __n)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`int __gnu_cxx::lexicographical_compare_3way (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2)`
- `template<typename _InputIterator, typename _RandomAccessIterator >`
`_RandomAccessIterator __gnu_cxx::random_sample (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out_first, _RandomAccessIterator __out_last)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _RandomNumberGenerator >`
`_RandomAccessIterator __gnu_cxx::random_sample (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __out_first, _RandomAccessIterator __out_last, _RandomNumberGenerator & __rand)`
- `template<typename _ForwardIterator, typename _OutputIterator, typename _Distance >`
`_OutputIterator __gnu_cxx::random_sample_n (_ForwardIterator __first, _ForwardIterator __last, _OutputIterator __out, const _Distance __n)`
- `template<typename _ForwardIterator, typename _OutputIterator, typename _Distance, typename _RandomNumberGenerator >`
`_OutputIterator __gnu_cxx::random_sample_n (_ForwardIterator __first, _ForwardIterator __last, _OutputIterator __out, const _Distance __n, _RandomNumberGenerator & __rand)`

6.4.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGI STL subset).

6.5 algorithm File Reference

Macros

- `#define _PARALLEL_ALGORITHM`

6.5.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.6 algorithm File Reference

Namespaces

- `std`

Macros

- `#define __cpp_lib_experimental_sample`
- `#define _GLIBCXX_EXPERIMENTAL_ALGORITHM`

Functions

- `template<typename _InputIterator, typename _RandomAccessIterator, typename _Size, typename _UniformRandomNumberGenerator >
_RandomAccessIterator std::experimental::fundamentals_v1::__sample (_InputIterator __first, _InputIterator __last, input_iterator_tag, _RandomAccessIterator __out, random_access_iterator_tag, _Size __n, _UniformRandomNumberGenerator &&__g)`
- `template<typename _ForwardIterator, typename _OutputIterator, typename _Cat, typename _Size, typename _UniformRandomNumberGenerator >
_OutputIterator std::experimental::fundamentals_v1::__sample (_ForwardIterator __first, _ForwardIterator __last, forward_iterator_tag, _OutputIterator __out, _Cat, _Size __n, _UniformRandomNumberGenerator &&__g)`
- `template<typename _PopulationIterator, typename _SampleIterator, typename _Distance, typename _UniformRandomNumberGenerator >
_SampleIterator std::experimental::fundamentals_v1::sample (_PopulationIterator __first, _PopulationIterator __last, _SampleIterator __out, _Distance __n, _UniformRandomNumberGenerator &&__g)`
- `template<typename _ForwardIterator, typename _Searcher >
_ForwardIterator std::experimental::fundamentals_v1::search (_ForwardIterator __first, _ForwardIterator __last, const _Searcher &__searcher)`

6.6.1 Detailed Description

This is a TS C++ Library header.

6.6.2 Function Documentation

6.6.2.1 `template<typename _InputIterator, typename _RandomAccessIterator, typename _Size, typename _UniformRandomNumberGenerator> _RandomAccessIterator std::experimental::fundamentals_v1::__sample (_InputIterator __first, _InputIterator __last, input_iterator_tag, _RandomAccessIterator __out, random_access_iterator_tag, _Size __n, _UniformRandomNumberGenerator && __g)`

Reservoir sampling algorithm.

Definition at line 62 of file experimental/algorithm.

6.6.2.2 `template<typename _ForwardIterator, typename _OutputIterator, typename _Cat, typename _Size, typename _UniformRandomNumberGenerator> _OutputIterator std::experimental::fundamentals_v1::__sample (_ForwardIterator __first, _ForwardIterator __last, forward_iterator_tag, _OutputIterator __out, _Cat, _Size __n, _UniformRandomNumberGenerator && __g)`

Selection sampling algorithm.

Definition at line 89 of file experimental/algorithm.

6.6.2.3 `template<typename _PopulationIterator, typename _SampleIterator, typename _Distance, typename _UniformRandomNumberGenerator> _SampleIterator std::experimental::fundamentals_v1::sample (_PopulationIterator __first, _PopulationIterator __last, _SampleIterator __out, _Distance __n, _UniformRandomNumberGenerator && __g)`

Take a random sample from a population.

Definition at line 111 of file experimental/algorithm.

6.7 algorithmfwd.h File Reference

Namespaces

- [std](#)

Functions

- `template<typename _Filter> _Filter std::adjacent_find (_Filter, _Filter)`
- `template<typename _Filter, typename _BinaryPredicate> _Filter std::adjacent_find (_Filter, _Filter, _BinaryPredicate)`
- `template<typename _Iter, typename _Predicate> bool std::all_of (_Iter, _Iter, _Predicate)`
- `template<typename _Iter, typename _Predicate> bool std::any_of (_Iter, _Iter, _Predicate)`
- `template<typename _Filter, typename _Tp> bool std::binary_search (_Filter, _Filter, const _Tp &)`
- `template<typename _Filter, typename _Tp, typename _Compare> bool std::binary_search (_Filter, _Filter, const _Tp &, _Compare)`

- `template<typename _Iter, typename _OIter >`
`_OIter std::copy (_Iter, _Iter, _OIter)`
- `template<typename _BIter1, typename _BIter2 >`
`_BIter2 std::copy_backward (_BIter1, _BIter1, _BIter2)`
- `template<typename _Iter, typename _OIter, typename _Predicate >`
`_OIter std::copy_if (_Iter, _Iter, _OIter, _Predicate)`
- `template<typename _Iter, typename _Size, typename _OIter >`
`_OIter std::copy_n (_Iter, _Size, _OIter)`
- `template<typename _Iter, typename _Tp >`
`iterator_traits< _Iter >::difference_type std::count (_Iter, _Iter, const _Tp &)`
- `template<typename _Iter, typename _Predicate >`
`iterator_traits< _Iter >::difference_type std::count_if (_Iter, _Iter, _Predicate)`
- `template<typename _Iter1, typename _Iter2 >`
`bool std::equal (_Iter1, _Iter1, _Iter2)`
- `template<typename _Iter1, typename _Iter2, typename _BinaryPredicate >`
`bool std::equal (_Iter1 __first1, _Iter1 __last1, _Iter2 __first2, _BinaryPredicate __binary_pred)`
- `template<typename _Filter, typename _Tp >`
`pair< _Filter, _Filter > std::equal_range (_Filter, _Filter, const _Tp &)`
- `template<typename _Filter, typename _Tp, typename _Compare >`
`pair< _Filter, _Filter > std::equal_range (_Filter, _Filter, const _Tp &, _Compare)`
- `template<typename _Filter, typename _Tp >`
`void std::fill (_Filter, _Filter, const _Tp &)`
- `template<typename _OIter, typename _Size, typename _Tp >`
`_OIter std::fill_n (_OIter, _Size, const _Tp &)`
- `template<typename _Iter, typename _Tp >`
`_Iter std::find (_Iter, _Iter, const _Tp &)`
- `template<typename _Filter1, typename _Filter2 >`
`_Filter1 std::find_end (_Filter1, _Filter1, _Filter2, _Filter2)`
- `template<typename _Filter1, typename _Filter2, typename _BinaryPredicate >`
`_Filter1 std::find_end (_Filter1, _Filter1, _Filter2, _Filter2, _BinaryPredicate)`
- `template<typename _Filter1, typename _Filter2 >`
`_Filter1 std::find_first_of (_Filter1, _Filter1, _Filter2, _Filter2)`
- `template<typename _Filter1, typename _Filter2, typename _BinaryPredicate >`
`_Filter1 std::find_first_of (_Filter1, _Filter1, _Filter2, _Filter2, _BinaryPredicate)`
- `template<typename _Iter, typename _Predicate >`
`_Iter std::find_if (_Iter, _Iter, _Predicate)`
- `template<typename _Iter, typename _Predicate >`
`_Iter std::find_if_not (_Iter, _Iter, _Predicate)`
- `template<typename _Iter, typename _Funct >`
`_Funct std::for_each (_Iter, _Iter, _Funct)`
- `template<typename _Filter, typename _Generator >`
`void std::generate (_Filter, _Filter, _Generator)`
- `template<typename _OIter, typename _Size, typename _Generator >`
`_OIter std::generate_n (_OIter, _Size, _Generator)`
- `template<typename _Iter1, typename _Iter2 >`
`bool std::includes (_Iter1, _Iter1, _Iter2, _Iter2)`
- `template<typename _Iter1, typename _Iter2, typename _Compare >`
`bool std::includes (_Iter1, _Iter1, _Iter2, _Iter2, _Compare)`
- `template<typename _BIter >`
`void std::inplace_merge (_BIter, _BIter, _BIter)`
- `template<typename _BIter, typename _Compare >`
`void std::inplace_merge (_BIter, _BIter, _BIter, _Compare)`

- `template<typename _RAIter >`
`bool std::is_heap (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`bool std::is_heap (_RAIter, _RAIter, _Compare)`
- `template<typename _RAIter >`
`_RAIter std::is_heap_until (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`_RAIter std::is_heap_until (_RAIter, _RAIter, _Compare)`
- `template<typename _Iter, typename _Predicate >`
`bool std::is_partitioned (_Iter, _Iter, _Predicate)`
- `template<typename _Filter1, typename _Filter2 >`
`bool std::is_permutation (_Filter1, _Filter1, _Filter2)`
- `template<typename _Filter1, typename _Filter2, typename _BinaryPredicate >`
`bool std::is_permutation (_Filter1, _Filter1, _Filter2, _BinaryPredicate)`
- `template<typename _Filter >`
`bool std::is_sorted (_Filter, _Filter)`
- `template<typename _Filter, typename _Compare >`
`bool std::is_sorted (_Filter, _Filter, _Compare)`
- `template<typename _Filter >`
`_Filter std::is_sorted_until (_Filter, _Filter)`
- `template<typename _Filter, typename _Compare >`
`_Filter std::is_sorted_until (_Filter, _Filter, _Compare)`
- `template<typename _Filter1, typename _Filter2 >`
`void std::iter_swap (_Filter1, _Filter2)`
- `template<typename _Iter1, typename _Iter2 >`
`bool std::lexicographical_compare (_Iter1, _Iter1, _Iter2, _Iter2)`
- `template<typename _Iter1, typename _Iter2, typename _Compare >`
`bool std::lexicographical_compare (_Iter1, _Iter1, _Iter2, _Iter2, _Compare)`
- `template<typename _Filter, typename _Tp >`
`_Filter std::lower_bound (_Filter, _Filter, const _Tp &)`
- `template<typename _Filter, typename _Tp, typename _Compare >`
`_Filter std::lower_bound (_Filter, _Filter, const _Tp &, _Compare)`
- `template<typename _RAIter >`
`void std::make_heap (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`void std::make_heap (_RAIter, _RAIter, _Compare)`
- `template<typename _Tp >`
`_GLIBCXX14_CONSTEXPR const _Tp & std::max (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp, typename _Compare >`
`_GLIBCXX14_CONSTEXPR const _Tp & std::max (const _Tp &__a, const _Tp &__b, _Compare __comp)`
- `template<typename _Tp >`
`_GLIBCXX14_CONSTEXPR _Tp std::max (initializer_list< _Tp >)`
- `template<typename _Tp, typename _Compare >`
`_GLIBCXX14_CONSTEXPR _Tp std::max (initializer_list< _Tp >, _Compare)`
- `template<typename _Filter >`
`_GLIBCXX14_CONSTEXPR _Filter std::max_element (_Filter, _Filter)`
- `template<typename _Filter, typename _Compare >`
`_GLIBCXX14_CONSTEXPR _Filter std::max_element (_Filter, _Filter, _Compare)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::merge (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`_OIter std::merge (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare)`

- `template<typename _Tp >`
`_GLIBCXX14_CONSTEXPR const _Tp & std::min (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp, typename _Compare >`
`_GLIBCXX14_CONSTEXPR const _Tp & std::min (const _Tp &__a, const _Tp &__b, _Compare __comp)`
- `template<typename _Tp >`
`_GLIBCXX14_CONSTEXPR _Tp std::min (initializer_list< _Tp >)`
- `template<typename _Tp, typename _Compare >`
`_GLIBCXX14_CONSTEXPR _Tp std::min (initializer_list< _Tp >, _Compare)`
- `template<typename _Filter >`
`_GLIBCXX14_CONSTEXPR _Filter std::min_element (_Filter, _Filter)`
- `template<typename _Filter, typename _Compare >`
`_GLIBCXX14_CONSTEXPR _Filter std::min_element (_Filter, _Filter, _Compare)`
- `template<typename _Tp >`
`_GLIBCXX14_CONSTEXPR pair< const _Tp &, const _Tp & > std::minmax (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp, typename _Compare >`
`_GLIBCXX14_CONSTEXPR pair< const _Tp &, const _Tp & > std::minmax (const _Tp &__a, const _Tp &__b, _Compare __comp)`
- `template<typename _Tp >`
`_GLIBCXX14_CONSTEXPR pair< _Tp, _Tp > std::minmax (initializer_list< _Tp >)`
- `template<typename _Tp, typename _Compare >`
`_GLIBCXX14_CONSTEXPR pair< _Tp, _Tp > std::minmax (initializer_list< _Tp >, _Compare)`
- `template<typename _Filter >`
`_GLIBCXX14_CONSTEXPR pair< _Filter, _Filter > std::minmax_element (_Filter, _Filter)`
- `template<typename _Filter, typename _Compare >`
`_GLIBCXX14_CONSTEXPR pair< _Filter, _Filter > std::minmax_element (_Filter, _Filter, _Compare)`
- `template<typename _IIter1, typename _IIter2 >`
`pair< _IIter1, _IIter2 > std::mismatch (_IIter1, _IIter1, _IIter2)`
- `template<typename _IIter1, typename _IIter2, typename _BinaryPredicate >`
`pair< _IIter1, _IIter2 > std::mismatch (_IIter1, _IIter1, _IIter2, _BinaryPredicate)`
- `template<typename _BIter >`
`bool std::next_permutation (_BIter, _BIter)`
- `template<typename _BIter, typename _Compare >`
`bool std::next_permutation (_BIter, _BIter, _Compare)`
- `template<typename _IIter, typename _Predicate >`
`bool std::none_of (_IIter, _IIter, _Predicate)`
- `template<typename _RAIter >`
`void std::nth_element (_RAIter, _RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`void std::nth_element (_RAIter, _RAIter, _RAIter, _Compare)`
- `template<typename _RAIter >`
`void std::partial_sort (_RAIter, _RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`void std::partial_sort (_RAIter, _RAIter, _RAIter, _Compare)`
- `template<typename _IIter, typename _RAIter >`
`_RAIter std::partial_sort_copy (_IIter, _IIter, _RAIter, _RAIter)`
- `template<typename _IIter, typename _RAIter, typename _Compare >`
`_RAIter std::partial_sort_copy (_IIter, _IIter, _RAIter, _RAIter, _Compare)`
- `template<typename _BIter, typename _Predicate >`
`_BIter std::partition (_BIter, _BIter, _Predicate)`
- `template<typename _IIter, typename _OIIter1, typename _OIIter2, typename _Predicate >`
`pair< _OIIter1, _OIIter2 > std::partition_copy (_IIter, _IIter, _OIIter1, _OIIter2, _Predicate)`
- `template<typename _Filter, typename _Predicate >`
`_Filter std::partition_point (_Filter, _Filter, _Predicate)`

- `template<typename _RAIter >`
`void std::pop_heap (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`void std::pop_heap (_RAIter, _RAIter, _Compare)`
- `template<typename _BIter >`
`bool std::prev_permutation (_BIter, _BIter)`
- `template<typename _BIter, typename _Compare >`
`bool std::prev_permutation (_BIter, _BIter, _Compare)`
- `template<typename _RAIter >`
`void std::push_heap (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`void std::push_heap (_RAIter, _RAIter, _Compare)`
- `template<typename _RAIter >`
`void std::random_shuffle (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Generator >`
`void std::random_shuffle (_RAIter, _RAIter, _Generator &&)`
- `template<typename _Filter, typename _Tp >`
`_Filter std::remove (_Filter, _Filter, const _Tp &)`
- `template<typename _Iter, typename _OIter, typename _Tp >`
`_OIter std::remove_copy (_Iter, _Iter, _OIter, const _Tp &)`
- `template<typename _Iter, typename _OIter, typename _Predicate >`
`_OIter std::remove_copy_if (_Iter, _Iter, _OIter, _Predicate)`
- `template<typename _Filter, typename _Predicate >`
`_Filter std::remove_if (_Filter, _Filter, _Predicate)`
- `template<typename _Filter, typename _Tp >`
`void std::replace (_Filter, _Filter, const _Tp &, const _Tp &)`
- `template<typename _Iter, typename _OIter, typename _Tp >`
`_OIter std::replace_copy (_Iter, _Iter, _OIter, const _Tp &, const _Tp &)`
- `template<typename _Iter, typename _OIter, typename _Predicate, typename _Tp >`
`_OIter std::replace_copy_if (_Iter, _Iter, _OIter, _Predicate, const _Tp &)`
- `template<typename _Filter, typename _Predicate, typename _Tp >`
`void std::replace_if (_Filter, _Filter, _Predicate, const _Tp &)`
- `template<typename _BIter >`
`void std::reverse (_BIter, _BIter)`
- `template<typename _BIter, typename _OIter >`
`_OIter std::reverse_copy (_BIter, _BIter, _OIter)`
- `template<typename _Filter >`
`_Filter std::V2::rotate (_Filter, _Filter, _Filter)`
- `template<typename _Filter, typename _OIter >`
`_OIter std::rotate_copy (_Filter, _Filter, _Filter, _OIter)`
- `template<typename _Filter1, typename _Filter2 >`
`_Filter1 std::search (_Filter1, _Filter1, _Filter2, _Filter2)`
- `template<typename _Filter1, typename _Filter2, typename _BinaryPredicate >`
`_Filter1 std::search (_Filter1, _Filter1, _Filter2, _Filter2, _BinaryPredicate)`
- `template<typename _Filter, typename _Size, typename _Tp >`
`_Filter std::search_n (_Filter, _Filter, _Size, const _Tp &)`
- `template<typename _Filter, typename _Size, typename _Tp, typename _BinaryPredicate >`
`_Filter std::search_n (_Filter, _Filter, _Size, const _Tp &, _BinaryPredicate)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::set_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`_OIter std::set_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare)`

- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::set_intersection (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`_OIter std::set_intersection (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::set_symmetric_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`_OIter std::set_symmetric_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`_OIter std::set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare)`
- `template<typename _RAIter, typename _UGenerator >`
`void std::shuffle (_RAIter, _RAIter, _UGenerator &&)`
- `template<typename _RAIter >`
`void std::sort (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`void std::sort (_RAIter, _RAIter, _Compare)`
- `template<typename _RAIter >`
`void std::sort_heap (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`void std::sort_heap (_RAIter, _RAIter, _Compare)`
- `template<typename _BIter, typename _Predicate >`
`_BIter std::stable_partition (_BIter, _BIter, _Predicate)`
- `template<typename _RAIter >`
`void std::stable_sort (_RAIter, _RAIter)`
- `template<typename _RAIter, typename _Compare >`
`void std::stable_sort (_RAIter, _RAIter, _Compare)`
- `template<typename _Filter1, typename _Filter2 >`
`_Filter2 std::swap_ranges (_Filter1, _Filter1, _Filter2)`
- `template<typename _Iter, typename _OIter, typename _UnaryOperation >`
`_OIter std::transform (_Iter, _Iter, _OIter, _UnaryOperation)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _BinaryOperation >`
`_OIter std::transform (_Iter1, _Iter1, _Iter2, _OIter, _BinaryOperation)`
- `template<typename _Filter >`
`_Filter std::unique (_Filter, _Filter)`
- `template<typename _Filter, typename _BinaryPredicate >`
`_Filter std::unique (_Filter, _Filter, _BinaryPredicate)`
- `template<typename _Iter, typename _OIter >`
`_OIter std::unique_copy (_Iter, _Iter, _OIter)`
- `template<typename _Iter, typename _OIter, typename _BinaryPredicate >`
`_OIter std::unique_copy (_Iter, _Iter, _OIter, _BinaryPredicate)`
- `template<typename _Filter, typename _Tp >`
`_Filter std::upper_bound (_Filter, _Filter, const _Tp &)`
- `template<typename _Filter, typename _Tp, typename _Compare >`
`_Filter std::upper_bound (_Filter, _Filter, const _Tp &, _Compare)`

6.7.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<algorithm>`.

6.8 algorithmfwd.h File Reference

Namespaces

- [std](#)
- [std::__parallel](#)

Functions

- `template<typename _Filter, typename _IterTag >
_Filter std::__parallel::__adjacent_find_switch (_Filter, _Filter, _IterTag)`
- `template<typename _Filter, typename _BiPredicate, typename _IterTag >
_Filter std::__parallel::__adjacent_find_switch (_Filter, _Filter, _BiPredicate, _IterTag)`
- `template<typename _RAIter, typename _BiPredicate >
_RAIter std::__parallel::__adjacent_find_switch (_RAIter, _RAIter, _BiPredicate, random_access_iterator_tag)`
- `template<typename _RAIter >
_RAIter std::__parallel::__adjacent_find_switch (_RAIter __begin, _RAIter __end, random_access_iterator_tag)`
- `template<typename _Iter, typename _Predicate, typename _IterTag >
iterator_traits< _Iter >::difference_type std::__parallel::__count_if_switch (_Iter, _Iter, _Predicate, _IterTag)`
- `template<typename _RAIter, typename _Predicate >
iterator_traits< _RAIter >::difference_type std::__parallel::__count_if_switch (_RAIter __begin, _RAIter __end, _Predicate __pred, random_access_iterator_tag, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Tp, typename _IterTag >
iterator_traits< _Iter >::difference_type std::__parallel::__count_switch (_Iter, _Iter, const _Tp &, _IterTag)`
- `template<typename _RAIter, typename _Tp >
iterator_traits< _RAIter >::difference_type std::__parallel::__count_switch (_RAIter __begin, _RAIter __end, const _Tp & __value, random_access_iterator_tag, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Filter, typename _IterTag1, typename _IterTag2 >
_Iter std::__parallel::__find_first_of_switch (_Iter, _Iter, _Filter, _Filter, _IterTag1, _IterTag2)`
- `template<typename _RAIter, typename _Filter, typename _BiPredicate, typename _IterTag >
_RAIter std::__parallel::__find_first_of_switch (_RAIter, _RAIter, _Filter, _Filter, _BiPredicate, random_access_iterator_tag, _IterTag)`
- `template<typename _Iter, typename _Filter, typename _BiPredicate, typename _IterTag1, typename _IterTag2 >
_Iter std::__parallel::__find_first_of_switch (_Iter, _Iter, _Filter, _Filter, _BiPredicate, _IterTag1, _IterTag2)`
- `template<typename _Iter, typename _Predicate, typename _IterTag >
_Iter std::__parallel::__find_if_switch (_Iter, _Iter, _Predicate, _IterTag)`
- `template<typename _RAIter, typename _Predicate >
_RAIter std::__parallel::__find_if_switch (_RAIter __begin, _RAIter __end, _Predicate __pred, random_access_iterator_tag)`
- `template<typename _RAIter, typename _Tp >
_RAIter std::__parallel::__find_switch (_RAIter __begin, _RAIter __end, const _Tp & __val, random_access_iterator_tag)`
- `template<typename _Iter, typename _Tp, typename _IterTag >
_Iter std::__parallel::__find_switch (_Iter, _Iter, const _Tp &, _IterTag)`
- `template<typename _RAIter, typename _Function >
_Function std::__parallel::__for_each_switch (_RAIter __begin, _RAIter __end, _Function __f, random_access_iterator_tag, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Function, typename _IterTag >
_Function std::__parallel::__for_each_switch (_Iter, _Iter, _Function, _IterTag)`

- `template<typename _OIter, typename _Size, typename _Generator, typename _IterTag >`
`_OIter std::parallel::generate_n_switch (_OIter, _Size, _Generator, _IterTag)`
- `template<typename _RAIter, typename _Size, typename _Generator >`
`_RAIter std::parallel::generate_n_switch (_RAIter __begin, _Size __n, _Generator __gen, random_↵`
`access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filter, typename _Generator, typename _IterTag >`
`void std::parallel::generate_switch (_Filter, _Filter, _Generator, _IterTag)`
- `template<typename _RAIter, typename _Generator >`
`void std::parallel::generate_switch (_RAIter __begin, _RAIter __end, _Generator __gen, random_↵`
`access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IterTag1, typename _IterTag2 >`
`bool std::parallel::lexicographical_compare_switch (_Iter1, _Iter1, _Iter2, _Iter2, _Predicate, _Iter↵`
`Tag1, _IterTag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _Predicate >`
`bool std::parallel::lexicographical_compare_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 ↵`
`__begin2, _RAIter2 __end2, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Filter, typename _Compare, typename _IterTag >`
`_Filter std::parallel::max_element_switch (_Filter, _Filter, _Compare, _IterTag)`
- `template<typename _RAIter, typename _Compare >`
`_RAIter std::parallel::max_element_switch (_RAIter __begin, _RAIter __end, _Compare __comp,`
`random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare, typename _IterTag1, typename _IterTag2, type-`
`name _IterTag3 >`
`_OIter std::parallel::merge_switch (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare, _IterTag1, _IterTag2,`
`_IterTag3)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`_OIter std::parallel::merge_switch (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare, random_access_↵`
`iterator_tag, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Filter, typename _Compare, typename _IterTag >`
`_Filter std::parallel::min_element_switch (_Filter, _Filter, _Compare, _IterTag)`
- `template<typename _RAIter, typename _Compare >`
`_RAIter std::parallel::min_element_switch (_RAIter __begin, _RAIter __end, _Compare __comp,`
`random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _RAIter1, typename _RAIter2, typename _Predicate >`
`pair< _RAIter1, _RAIter2 > std::parallel::mismatch_switch (_RAIter1 __begin1, _RAIter1 __end1, _R↵`
`Alter2 __begin2, _Predicate __pred, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _IterTag1, typename _IterTag2 >`
`pair< _Iter1, _Iter2 > std::parallel::mismatch_switch (_Iter1, _Iter1, _Iter2, _Predicate, _IterTag1, ↵`
`IterTag2)`
- `template<typename _Filter, typename _Predicate, typename _IterTag >`
`_Filter std::parallel::partition_switch (_Filter, _Filter, _Predicate, _IterTag)`
- `template<typename _RAIter, typename _Predicate >`
`_RAIter std::parallel::partition_switch (_RAIter __begin, _RAIter __end, _Predicate __pred, random_↵`
`access_iterator_tag)`
- `template<typename _Filter, typename _Predicate, typename _Tp, typename _IterTag >`
`void std::parallel::replace_if_switch (_Filter, _Filter, _Predicate, const _Tp &, _IterTag)`
- `template<typename _RAIter, typename _Predicate, typename _Tp >`
`void std::parallel::replace_if_switch (_RAIter __begin, _RAIter __end, _Predicate __pred, const _Tp &↵`
`__new_value, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filter, typename _Tp, typename _IterTag >`
`void std::parallel::replace_switch (_Filter, _Filter, const _Tp &, const _Tp &, _IterTag)`
- `template<typename _RAIter, typename _Tp >`
`void std::parallel::replace_switch (_RAIter __begin, _RAIter __end, const _Tp & __old_value, const _Tp`
`& __new_value, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism_tag)`

- `template<typename _RAIter, typename _Integer, typename _Tp, typename _BiPredicate >`
`_RAIter std::parallel::search_n_switch (_RAIter, _RAIter, _Integer, const _Tp &, _BiPredicate, random_↵`
`_access_iterator_tag)`
- `template<typename _Filter, typename _Integer, typename _Tp, typename _BiPredicate, typename _IterTag >`
`_Filter std::parallel::search_n_switch (_Filter, _Filter, _Integer, const _Tp &, _BiPredicate, _IterTag)`
- `template<typename _Filter1, typename _Filter2, typename _IterTag1, typename _IterTag2 >`
`_Filter1 std::parallel::search_switch (_Filter1, _Filter1, _Filter2, _Filter2, _IterTag1, _IterTag2)`
- `template<typename _RAIter1, typename _RAIter2, typename _BiPredicate >`
`_RAIter1 std::parallel::search_switch (_RAIter1, _RAIter1, _RAIter2, _RAIter2, _BiPredicate, random_↵`
`access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Filter1, typename _Filter2, typename _BiPredicate, typename _IterTag1, typename _IterTag2 >`
`_Filter1 std::parallel::search_switch (_Filter1, _Filter1, _Filter2, _Filter2, _BiPredicate, _IterTag1, _IterTag2)`
- `template<typename _RAIter1, typename _RAIter2 >`
`_RAIter1 std::parallel::search_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _RAIter2`
`__end2, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _RAIter1, typename _RAIter2, typename _Output_RAlter, typename _Predicate >`
`_Output_RAlter std::parallel::set_difference_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 ↵`
`__begin2, _RAIter2 __end2, _Output_RAlter __result, _Predicate __pred, random_access_iterator_tag, random_↵`
`_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _Olter, typename _IterTag1, typename _IterTag2, type-`
`name _IterTag3 >`
`_Olter std::parallel::set_difference_switch (_Iter1, _Iter1, _Iter2, _Iter2, _Olter, _Predicate, _IterTag1,`
`_IterTag2, _IterTag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _Output_RAlter, typename _Predicate >`
`_Output_RAlter std::parallel::set_intersection_switch (_RAIter1 __begin1, _RAIter1 __end1, _RA↵`
`Iter2 __begin2, _RAIter2 __end2, _Output_RAlter __result, _Predicate __pred, random_access_iterator_tag,`
`random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _Olter, typename _IterTag1, typename _IterTag2, type-`
`name _IterTag3 >`
`_Olter std::parallel::set_intersection_switch (_Iter1, _Iter1, _Iter2, _Iter2, _Olter, _Predicate, _Iter↵`
`Tag1, _IterTag2, _IterTag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _Output_RAlter, typename _Predicate >`
`_Output_RAlter std::parallel::set_symmetric_difference_switch (_RAIter1 __begin1, _RAIter1 __end1,`
`_RAIter2 __begin2, _RAIter2 __end2, _Output_RAlter __result, _Predicate __pred, random_access_iterator_↵`
`tag, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _Olter, typename _IterTag1, typename _IterTag2 ,`
`typename _IterTag3 >`
`_Olter std::parallel::set_symmetric_difference_switch (_Iter1, _Iter1, _Iter2, _Iter2, _Olter, ↵`
`Predicate, _IterTag1, _IterTag2, _IterTag3)`
- `template<typename _RAIter1, typename _RAIter2, typename _Output_RAlter, typename _Predicate >`
`_Output_RAlter std::parallel::set_union_switch (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 ↵`
`begin2, _RAIter2 __end2, _Output_RAlter __result, _Predicate __pred, random_access_iterator_tag, random_↵`
`_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate, typename _Olter, typename _IterTag1, typename _IterTag2, type-`
`name _IterTag3 >`
`_Olter std::parallel::set_union_switch (_Iter1, _Iter1, _Iter2, _Iter2, _Olter, ↵`
`Predicate, _IterTag1, _IterTag2, _IterTag3)`
- `template<typename _Iter, typename _Olter, typename _UnaryOperation, typename _IterTag1, typename _IterTag2 >`
`_Olter std::parallel::transform1_switch (_Iter, _Iter, _Olter, _UnaryOperation, _IterTag1, _IterTag2)`
- `template<typename _RAIter, typename _RAOlter, typename _UnaryOperation >`
`_RAOlter std::parallel::transform1_switch (_RAIter, _RAIter, _RAOlter, _UnaryOperation, random_↵`
`access_iterator_tag, random_access_iterator_tag, gnu_parallel::Parallelism __parallelism=gnu_parallel_↵`
`::parallel_balanced)`

- `template<typename _RAIter1, typename _RAIter2, typename _RAIter3, typename _BiOperation >`
`_RAIter3 std::parallel::transform2_switch (_RAIter1, _RAIter1, _RAIter2, _RAIter3, _BiOperation,`
`random_access_iterator_tag, random_access_iterator_tag, random_access_iterator_tag, __gnu_parallel::Parallelism __parallelism=__gnu_parallel::parallel_balanced)`
- `template<typename _IIter1, typename _IIter2, typename _OIter, typename _BiOperation, typename _Tag1, typename _Tag2, typename`
`_Tag3 >`
`_OIter std::parallel::transform2_switch (_IIter1, _IIter1, _IIter2, _OIter, _BiOperation, _Tag1, _Tag2, __gnu_parallel::parallel_balanced)`
- `template<typename _IIter, typename _OIter, typename _Predicate, typename _IterTag1, typename _IterTag2 >`
`_OIter std::parallel::unique_copy_switch (_IIter, _IIter, _OIter, _Predicate, _IterTag1, _IterTag2)`
- `template<typename _RAIter, typename _RandomAccess_OIter, typename _Predicate >`
`_RandomAccess_OIter std::parallel::unique_copy_switch (_RAIter, _RAIter, _RandomAccess_OIter, __gnu_parallel::parallel_balanced,`
`Predicate, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Filter >`
`_Filter std::parallel::adjacent_find (_Filter, _Filter)`
- `template<typename _Filter >`
`_Filter std::parallel::adjacent_find (_Filter, _Filter, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _BiPredicate >`
`_Filter std::parallel::adjacent_find (_Filter, _Filter, _BiPredicate)`
- `template<typename _Filter, typename _BiPredicate >`
`_Filter std::parallel::adjacent_find (_Filter, _Filter, _BiPredicate, __gnu_parallel::sequential_tag)`
- `template<typename _IIter, typename _Tp >`
`iterator_traits< _IIter >::difference_type std::parallel::count (_IIter __begin, _IIter __end, const _Tp & __gnu_parallel::sequential_tag`
`value, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _IIter, typename _Tp >`
`iterator_traits< _IIter >::difference_type std::parallel::count (_IIter __begin, _IIter __end, const _Tp & __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _IIter, typename _Tp >`
`iterator_traits< _IIter >::difference_type std::parallel::count (_IIter __begin, _IIter __end, const _Tp & __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _IIter, typename _Predicate >`
`iterator_traits< _IIter >::difference_type std::parallel::count_if (_IIter __begin, _IIter __end, _Predicate __gnu_parallel::sequential_tag`
`pred, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _IIter, typename _Predicate >`
`iterator_traits< _IIter >::difference_type std::parallel::count_if (_IIter __begin, _IIter __end, _Predicate __gnu_parallel::sequential_tag`
`pred, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _IIter1, typename _IIter2 >`
`bool std::parallel::equal (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _Predicate >`
`bool std::parallel::equal (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2 >`
`bool std::parallel::equal (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2)`
- `template<typename _IIter1, typename _IIter2, typename _Predicate >`
`bool std::parallel::equal (_IIter1 __begin1, _IIter1 __end1, _IIter2 __begin2, _Predicate __pred)`
- `template<typename _IIter, typename _Tp >`
`_IIter std::parallel::find (_IIter __begin, _IIter __end, const _Tp & __val, __gnu_parallel::sequential_tag)`
- `template<typename _IIter, typename _Tp >`
`_IIter std::parallel::find (_IIter __begin, _IIter __end, const _Tp & __val)`

- `template<typename _Iter, typename _Filter >`
`_Iter std::parallel::find_first_of (_Iter, _Iter, _Filter, _Filter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Filter, typename _BiPredicate >`
`_Iter std::parallel::find_first_of (_Iter, _Iter, _Filter, _Filter, _BiPredicate, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Filter, typename _BiPredicate >`
`_Iter std::parallel::find_first_of (_Iter, _Iter, _Filter, _Filter, _BiPredicate)`
- `template<typename _Iter, typename _Filter >`
`_Iter std::parallel::find_first_of (_Iter, _Iter, _Filter, _Filter)`
- `template<typename _Iter, typename _Predicate >`
`_Iter std::parallel::find_if (_Iter __begin, _Iter __end, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Predicate >`
`_Iter std::parallel::find_if (_Iter __begin, _Iter __end, _Predicate __pred)`
- `template<typename _Iter, typename _Function >`
`_Function std::parallel::for_each (_Iter __begin, _Iter __end, _Function __f, __gnu_parallel::sequential_tag)`
- `template<typename _Iterator, typename _Function >`
`_Function std::parallel::for_each (_Iterator __begin, _Iterator __end, _Function __f, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Function >`
`_Function std::parallel::for_each (_Iter, _Iter, _Function)`
- `template<typename _Filter, typename _Generator >`
`void std::parallel::generate (_Filter, _Filter, _Generator)`
- `template<typename _Filter, typename _Generator >`
`void std::parallel::generate (_Filter, _Filter, _Generator, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Generator >`
`void std::parallel::generate (_Filter, _Filter, _Generator, __gnu_parallel::Parallelism)`
- `template<typename _OIter, typename _Size, typename _Generator >`
`_OIter std::parallel::generate_n (_OIter, _Size, _Generator)`
- `template<typename _OIter, typename _Size, typename _Generator >`
`_OIter std::parallel::generate_n (_OIter, _Size, _Generator, __gnu_parallel::sequential_tag)`
- `template<typename _OIter, typename _Size, typename _Generator >`
`_OIter std::parallel::generate_n (_OIter, _Size, _Generator, __gnu_parallel::Parallelism)`
- `template<typename _Iter1, typename _Iter2 >`
`bool std::parallel::lexicographical_compare (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`bool std::parallel::lexicographical_compare (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`bool std::parallel::lexicographical_compare (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`bool std::parallel::lexicographical_compare (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, _Iter2 __end2, _Predicate __pred)`
- `template<typename _Filter >`
`_Filter std::parallel::max_element (_Filter, _Filter)`
- `template<typename _Filter >`
`_Filter std::parallel::max_element (_Filter, _Filter, __gnu_parallel::sequential_tag)`
- `template<typename _Filter >`
`_Filter std::parallel::max_element (_Filter, _Filter, __gnu_parallel::Parallelism)`
- `template<typename _Filter, typename _Compare >`
`_Filter std::parallel::max_element (_Filter, _Filter, _Compare)`
- `template<typename _Filter, typename _Compare >`
`_Filter std::parallel::max_element (_Filter, _Filter, _Compare, __gnu_parallel::sequential_tag)`

- `template<typename _Filter, typename _Compare >`
`_Filter std::parallel::max_element (_Filter, _Filter, _Compare, __gnu_parallel::Parallelism)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::parallel::merge (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`_OIter std::parallel::merge (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Compare >`
`_OIter std::parallel::merge (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Compare)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::parallel::merge (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Filter >`
`_Filter std::parallel::min_element (_Filter, _Filter)`
- `template<typename _Filter >`
`_Filter std::parallel::min_element (_Filter, _Filter, __gnu_parallel::sequential_tag)`
- `template<typename _Filter >`
`_Filter std::parallel::min_element (_Filter, _Filter, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Filter, typename _Compare >`
`_Filter std::parallel::min_element (_Filter, _Filter, _Compare)`
- `template<typename _Filter, typename _Compare >`
`_Filter std::parallel::min_element (_Filter, _Filter, _Compare, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Compare >`
`_Filter std::parallel::min_element (_Filter, _Filter, _Compare, __gnu_parallel::Parallelism)`
- `template<typename _Iter1, typename _Iter2 >`
`pair< _Iter1, _Iter2 > std::parallel::mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`pair< _Iter1, _Iter2 > std::parallel::mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, __gnu_parallel::sequential_tag, _Predicate __pred, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2 >`
`pair< _Iter1, _Iter2 > std::parallel::mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2)`
- `template<typename _Iter1, typename _Iter2, typename _Predicate >`
`pair< _Iter1, _Iter2 > std::parallel::mismatch (_Iter1 __begin1, _Iter1 __end1, _Iter2 __begin2, __gnu_parallel::sequential_tag, _Predicate __pred)`
- `template<typename _RAlter >`
`void std::parallel::nth_element (_RAlter __begin, _RAlter __nth, _RAlter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAlter, typename _Compare >`
`void std::parallel::nth_element (_RAlter __begin, _RAlter __nth, _RAlter __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _RAlter, typename _Compare >`
`void std::parallel::nth_element (_RAlter __begin, _RAlter __nth, _RAlter __end, _Compare __comp)`
- `template<typename _RAlter >`
`void std::parallel::nth_element (_RAlter __begin, _RAlter __nth, _RAlter __end)`
- `template<typename _RAlter, typename _Compare >`
`void std::parallel::partial_sort (_RAlter __begin, _RAlter __middle, _RAlter __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _RAlter >`
`void std::parallel::partial_sort (_RAlter __begin, _RAlter __middle, _RAlter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAlter, typename _Compare >`
`void std::parallel::partial_sort (_RAlter __begin, _RAlter __middle, _RAlter __end, _Compare __comp)`

- `template<typename _RAIter >`
`void std::__parallel::partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end)`
- `template<typename _Filter, typename _Predicate >`
`_Filter std::__parallel::partition (_Filter, _Filter, _Predicate, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Predicate >`
`_Filter std::__parallel::partition (_Filter, _Filter, _Predicate)`
- `template<typename _RAIter >`
`void std::__parallel::random_shuffle (_RAIter __begin, _RAIter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _RandomNumberGenerator >`
`void std::__parallel::random_shuffle (_RAIter __begin, _RAIter __end, _RandomNumberGenerator &__rand, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter >`
`void std::__parallel::random_shuffle (_RAIter __begin, _RAIter __end)`
- `template<typename _RAIter, typename _RandomNumberGenerator >`
`void std::__parallel::random_shuffle (_RAIter __begin, _RAIter __end, _RandomNumberGenerator &&__rand)`
- `template<typename _Filter, typename _Tp >`
`void std::__parallel::replace (_Filter, _Filter, const _Tp &, const _Tp &)`
- `template<typename _Filter, typename _Tp >`
`void std::__parallel::replace (_Filter, _Filter, const _Tp &, const _Tp &, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Tp >`
`void std::__parallel::replace (_Filter, _Filter, const _Tp &, const _Tp &, __gnu_parallel::Parallelism)`
- `template<typename _Filter, typename _Predicate, typename _Tp >`
`void std::__parallel::replace_if (_Filter, _Filter, _Predicate, const _Tp &)`
- `template<typename _Filter, typename _Predicate, typename _Tp >`
`void std::__parallel::replace_if (_Filter, _Filter, _Predicate, const _Tp &, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Predicate, typename _Tp >`
`void std::__parallel::replace_if (_Filter, _Filter, _Predicate, const _Tp &, __gnu_parallel::Parallelism)`
- `template<typename _Filter1, typename _Filter2 >`
`_Filter1 std::__parallel::search (_Filter1, _Filter1, _Filter2, _Filter2, __gnu_parallel::sequential_tag)`
- `template<typename _Filter1, typename _Filter2 >`
`_Filter1 std::__parallel::search (_Filter1, _Filter1, _Filter2, _Filter2)`
- `template<typename _Filter1, typename _Filter2, typename _BiPredicate >`
`_Filter1 std::__parallel::search (_Filter1, _Filter1, _Filter2, _Filter2, _BiPredicate, __gnu_parallel::sequential_tag)`
- `template<typename _Filter1, typename _Filter2, typename _BiPredicate >`
`_Filter1 std::__parallel::search (_Filter1, _Filter1, _Filter2, _Filter2, _BiPredicate)`
- `template<typename _Filter, typename _Integer, typename _Tp >`
`_Filter std::__parallel::search_n (_Filter, _Filter, _Integer, const _Tp &, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Integer, typename _Tp, typename _BiPredicate >`
`_Filter std::__parallel::search_n (_Filter, _Filter, _Integer, const _Tp &, _BiPredicate, __gnu_parallel::sequential_tag)`
- `template<typename _Filter, typename _Integer, typename _Tp >`
`_Filter std::__parallel::search_n (_Filter, _Filter, _Integer, const _Tp &)`
- `template<typename _Filter, typename _Integer, typename _Tp, typename _BiPredicate >`
`_Filter std::__parallel::search_n (_Filter, _Filter, _Integer, const _Tp &, _BiPredicate)`
- `template<typename _IIter1, typename _IIter2, typename _OIter >`
`_OIter std::__parallel::set_difference (_IIter1, _IIter1, _IIter2, _IIter2, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _OIter, typename _Predicate >`
`_OIter std::__parallel::set_difference (_IIter1, _IIter1, _IIter2, _IIter2, _OIter, _Predicate, __gnu_parallel::sequential_tag)`
- `template<typename _IIter1, typename _IIter2, typename _OIter >`
`_OIter std::__parallel::set_difference (_IIter1, _IIter1, _IIter2, _IIter2, _OIter)`
- `template<typename _IIter1, typename _IIter2, typename _OIter, typename _Predicate >`
`_OIter std::__parallel::set_difference (_IIter1, _IIter1, _IIter2, _IIter2, _OIter, _Predicate)`

- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::__parallel::set_intersection (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter std::__parallel::set_intersection (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::__parallel::set_intersection (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter std::__parallel::set_intersection (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::__parallel::set_symmetric_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter std::__parallel::set_symmetric_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::__parallel::set_symmetric_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter std::__parallel::set_symmetric_difference (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::__parallel::set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter std::__parallel::set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter >`
`_OIter std::__parallel::set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _Predicate >`
`_OIter std::__parallel::set_union (_Iter1, _Iter1, _Iter2, _Iter2, _OIter, _Predicate)`
- `template<typename _RAIter >`
`void std::__parallel::sort (_RAIter __begin, _RAIter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::sort (_RAIter __begin, _RAIter __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter >`
`void std::__parallel::sort (_RAIter __begin, _RAIter __end)`
- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::sort (_RAIter __begin, _RAIter __end, _Compare __comp)`
- `template<typename _RAIter >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _RAIter >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end)`
- `template<typename _RAIter, typename _Compare >`
`void std::__parallel::stable_sort (_RAIter __begin, _RAIter __end, _Compare __comp)`
- `template<typename _Iter, typename _OIter, typename _UnaryOperation >`
`_OIter std::__parallel::transform (_Iter, _Iter, _OIter, _UnaryOperation)`
- `template<typename _Iter, typename _OIter, typename _UnaryOperation >`
`_OIter std::__parallel::transform (_Iter, _Iter, _OIter, _UnaryOperation, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter, typename _UnaryOperation >`
`_OIter std::__parallel::transform (_Iter, _Iter, _OIter, _UnaryOperation, __gnu_parallel::Parallelism)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _BiOperation >`
`_OIter std::__parallel::transform (_Iter1, _Iter1, _Iter2, _OIter, _BiOperation)`

- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _BiOperation >
_OIter std::__parallel::transform (_Iter1, _Iter1, _Iter2, _OIter, _BiOperation, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _OIter, typename _BiOperation >
_OIter std::__parallel::transform (_Iter1, _Iter1, _Iter2, _OIter, _BiOperation, __gnu_parallel::_Parallelism)`
- `template<typename _Iter, typename _OIter >
_OIter std::__parallel::unique_copy (_Iter, _Iter, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter, typename _Predicate >
_OIter std::__parallel::unique_copy (_Iter, _Iter, _OIter, _Predicate, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter >
_OIter std::__parallel::unique_copy (_Iter, _Iter, _OIter)`
- `template<typename _Iter, typename _OIter, typename _Predicate >
_OIter std::__parallel::unique_copy (_Iter, _Iter, _OIter, _Predicate)`

6.8.1 Detailed Description

This file is a GNU parallel extension to the Standard C++ Library.

6.9 aligned_buffer.h File Reference

Namespaces

- [__gnu_cxx](#)

6.9.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.10 alloc_traits.h File Reference

Classes

- struct [std::allocator_traits<_Alloc>](#)
- struct [std::allocator_traits< allocator<_Tp>>](#)

Namespaces

- [std](#)

Macros

- `#define __cpp_lib_allocator_traits_is_always_equal`

Typedefs

- `template<typename _Alloc, typename _Up >`
`using std::__alloc_rebind = __detected_or_t< __replace_first_arg_t, __allocator_traits_base::__rebind, _↔`
`_Alloc, _Up >`

Functions

- `template<typename _Alloc >`
`void std::__alloc_on_copy (_Alloc &__one, const _Alloc &__two)`
- `template<typename _Alloc >`
`_Alloc std::__alloc_on_copy (const _Alloc &__a)`
- `template<typename _Alloc >`
`void std::__alloc_on_move (_Alloc &__one, _Alloc &__two)`
- `template<typename _Alloc >`
`void std::__alloc_on_swap (_Alloc &__one, _Alloc &__two)`
- `template<typename _Alloc >`
`void std::__do_alloc_on_copy (_Alloc &__one, const _Alloc &__two, true_type)`
- `template<typename _Alloc >`
`void std::__do_alloc_on_copy (_Alloc &, const _Alloc &, false_type)`
- `template<typename _Alloc >`
`void std::__do_alloc_on_move (_Alloc &__one, _Alloc &__two, true_type)`
- `template<typename _Alloc >`
`void std::__do_alloc_on_move (_Alloc &, _Alloc &, false_type)`
- `template<typename _Alloc >`
`void std::__do_alloc_on_swap (_Alloc &__one, _Alloc &__two, true_type)`
- `template<typename _Alloc >`
`void std::__do_alloc_on_swap (_Alloc &, _Alloc &, false_type)`

6.10.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

6.11 `alloc_traits.h` File Reference

Classes

- `struct __gnu_cxx::__alloc_traits< _Alloc >`

Namespaces

- `__gnu_cxx`

6.11.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.12 `allocated_ptr.h` File Reference

Classes

- struct `std::__allocated_ptr<_Alloc>`

Namespaces

- `std`

Functions

- template<typename `_Alloc`>
`__allocated_ptr<_Alloc> std::__allocate_guarded` (`_Alloc &__a`)

6.12.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

6.13 `allocator.h` File Reference

Classes

- class `std::allocator<_Tp>`
- class `std::allocator<void>`

Namespaces

- `std`

Macros

- `#define __cpp_lib_allocator_is_always_equal`
- `#define __cpp_lib_incomplete_container_elements`

Functions

- template<typename `_T1`, typename `_T2`>
bool **`std::operator!=`** (const allocator< `_T1` > &, const allocator< `_T2` > &) noexcept
- template<typename `_Tp`>
bool **`std::operator!=`** (const allocator< `_Tp` > &, const allocator< `_Tp` > &) noexcept
- template<typename `_T1`, typename `_T2`>
bool **`std::operator==`** (const allocator< `_T1` > &, const allocator< `_T2` > &) noexcept
- template<typename `_Tp`>
bool **`std::operator==`** (const allocator< `_Tp` > &, const allocator< `_Tp` > &) noexcept

6.13.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

6.14 any File Reference

Classes

- class [std::experimental::fundamentals_v1::any](#)
- class [std::experimental::fundamentals_v1::bad_any_cast](#)

Namespaces

- [std](#)

Macros

- `#define __cpp_lib_experimental_any`
- `#define _GLIBCXX_EXPERIMENTAL_ANY`

Functions

- `template<typename _Tp >`
`void * std::experimental::fundamentals_v1::__any_caster (const any *__any)`
- `void std::experimental::fundamentals_v1::__throw_bad_any_cast ()`
- `template<typename _ValueType >`
`_ValueType std::experimental::fundamentals_v1::any_cast (const any &__any)`
- `void std::experimental::fundamentals_v1::swap (any &__x, any &__y) noexcept`
- `template<typename _ValueType >`
`_ValueType std::experimental::fundamentals_v1::any_cast (any &__any)`
- `template<typename _ValueType , typename enable_if<!is_move_constructible< _ValueType >::value || is_lvalue_reference< _ValueType >::value, bool >::type = true>`
`_ValueType std::experimental::fundamentals_v1::any_cast (any &&__any)`
- `template<typename _ValueType >`
`const _ValueType * std::experimental::fundamentals_v1::any_cast (const any *__any) noexcept`
- `template<typename _ValueType >`
`_ValueType * std::experimental::fundamentals_v1::any_cast (any *__any) noexcept`

6.14.1 Detailed Description

This is a TS C++ Library header.

6.15 array File Reference

Classes

- struct [std::array<_Tp, _Nm>](#)
- struct [std::tuple_element<_Int, _Tp>](#)
- struct [std::tuple_element<_Int, ::array<_Tp, _Nm>>](#)
- struct [std::tuple_size<_Tp>](#)
- struct [std::tuple_size<::array<_Tp, _Nm>>](#)

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_ARRAY`

Functions

- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>
constexpr _Tp & std::get (array< _Tp, _Nm > &__arr) noexcept`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>
constexpr _Tp && std::get (array< _Tp, _Nm > &&__arr) noexcept`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>
constexpr const _Tp & std::get (const array< _Tp, _Nm > &__arr) noexcept`
- `template<typename _Tp, std::size_t _Nm>
bool std::operator!= (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _Tp, std::size_t _Nm>
bool std::operator< (const array< _Tp, _Nm > &__a, const array< _Tp, _Nm > &__b)`
- `template<typename _Tp, std::size_t _Nm>
bool std::operator<= (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _Tp, std::size_t _Nm>
bool std::operator== (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _Tp, std::size_t _Nm>
bool std::operator> (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _Tp, std::size_t _Nm>
bool std::operator>= (const array< _Tp, _Nm > &__one, const array< _Tp, _Nm > &__two)`
- `template<typename _Tp, std::size_t _Nm>
void std::swap (array< _Tp, _Nm > &__one, array< _Tp, _Nm > &__two) noexcept(noexcept(__one.swap(__two)))`

6.15.1 Detailed Description

This is a Standard C++ Library header.

6.16 array File Reference

Classes

- struct [std::tuple_element<_Int, std::__debug::array<_Tp, _Nm>>](#)
- struct [std::tuple_size<std::__debug::array<_Tp, _Nm>>](#)

Namespaces

- [std](#)
- [std::__debug](#)

Macros

- `#define _GLIBCXX_DEBUG_ARRAY`

Functions

- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>
constexpr _Tp & std::__debug::get (array<_Tp, _Nm> &__arr) noexcept`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>
constexpr _Tp && std::__debug::get (array<_Tp, _Nm> &&__arr) noexcept`
- `template<std::size_t _Int, typename _Tp, std::size_t _Nm>
constexpr const _Tp & std::__debug::get (const array<_Tp, _Nm> &__arr) noexcept`
- `template<typename _Tp, std::size_t _Nm>
bool std::__debug::operator!= (const array<_Tp, _Nm> &__one, const array<_Tp, _Nm> &__two)`
- `template<typename _Tp, std::size_t _Nm>
bool std::__debug::operator< (const array<_Tp, _Nm> &__a, const array<_Tp, _Nm> &__b)`
- `template<typename _Tp, std::size_t _Nm>
bool std::__debug::operator<= (const array<_Tp, _Nm> &__one, const array<_Tp, _Nm> &__two)`
- `template<typename _Tp, std::size_t _Nm>
bool std::__debug::operator== (const array<_Tp, _Nm> &__one, const array<_Tp, _Nm> &__two)`
- `template<typename _Tp, std::size_t _Nm>
bool std::__debug::operator> (const array<_Tp, _Nm> &__one, const array<_Tp, _Nm> &__two)`
- `template<typename _Tp, std::size_t _Nm>
bool std::__debug::operator>= (const array<_Tp, _Nm> &__one, const array<_Tp, _Nm> &__two)`
- `template<typename _Tp, std::size_t _Nm>
void std::__debug::swap (array<_Tp, _Nm> &__one, array<_Tp, _Nm> &__two) noexcept(noexcept(__↵
one.swap(__two)))`

6.16.1 Detailed Description

This is a Standard C++ Library header.

6.17 array File Reference

Namespaces

- [std](#)

Macros

- `#define __cpp_lib_experimental_make_array`
- `#define _GLIBCXX_EXPERIMENTAL_ARRAY`

Functions

- `template<typename _Tp, size_t _Nm, size_t... _Idx>
constexpr array< remove_cv_t< _Tp >, _Nm > std::experimental::fundamentals_v2::__to_array (_Tp(&__↵
_a)[_Nm], index_sequence< _Idx... >)`
- `template<typename _Dest = void, typename... _Types>
constexpr auto std::experimental::fundamentals_v2::make_array (_Types &&... __t) -> array< conditional_t<
is_void_v< _Dest >, common_type_t< _Types... >, _Dest >, sizeof...(_Types)>`
- `template<typename _Tp, size_t _Nm>
constexpr array< remove_cv_t< _Tp >, _Nm > std::experimental::fundamentals_v2::to_array (_Tp(&__↵
a)[_Nm])`

6.17.1 Detailed Description

This is a TS C++ Library header.

6.18 array_allocator.h File Reference

Classes

- class [__gnu_cxx::array_allocator< _Tp, _Array >](#)
- class [__gnu_cxx::array_allocator_base< _Tp >](#)

Namespaces

- [__gnu_cxx](#)

Functions

- `template<typename _Tp, typename _Array >
bool __gnu_cxx::operator!= (const array_allocator< _Tp, _Array > &, const array_allocator< _Tp, _Array > &)`
- `template<typename _Tp, typename _Array >
bool __gnu_cxx::operator== (const array_allocator< _Tp, _Array > &, const array_allocator< _Tp, _Array > &)`

6.18.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.19 assertions.h File Reference

Macros

- `#define _GLIBCXX_DEBUG_ASSERT(_Condition)`
- `#define _GLIBCXX_DEBUG_ONLY(_Statement)`
- `#define _GLIBCXX_DEBUG_PEDASSERT(_Condition)`

6.19.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.20 assoc_container.hpp File Reference

Classes

- class [__gnu_pbds::basic_branch](#)< Key, Mapped, Tag, Node_Update, Policy_Tl, _Alloc >
- class [__gnu_pbds::basic_hash_table](#)< Key, Mapped, Hash_Fn, Eq_Fn, Resize_Policy, Store_Hash, Tag, Policy_Tl, _Alloc >
- class [__gnu_pbds::cc_hash_table](#)< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy, Store_↵ Hash, _Alloc >
- class [__gnu_pbds::gp_hash_table](#)< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_↵ Policy, Store_Hash, _Alloc >
- class [__gnu_pbds::list_update](#)< Key, Mapped, Eq_Fn, Update_Policy, _Alloc >
- class [__gnu_pbds::tree](#)< Key, Mapped, Cmp_Fn, Tag, Node_Update, _Alloc >
- class [__gnu_pbds::trie](#)< Key, Mapped, _ATraits, Tag, Node_Update, _Alloc >

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_BRANCH_BASE`
- `#define PB_DS_CC_HASH_BASE`
- `#define PB_DS_GP_HASH_BASE`
- `#define PB_DS_HASH_BASE`
- `#define PB_DS_LU_BASE`
- `#define PB_DS_TREE_BASE`
- `#define PB_DS_TREE_NODE_AND_IT_TRAITS`
- `#define PB_DS_TRIE_BASE`
- `#define PB_DS_TRIE_NODE_AND_IT_TRAITS`

6.20.1 Detailed Description

Contains associative containers.

6.21 atomic File Reference

Classes

- struct [std::atomic< _Tp >](#)
- struct [std::atomic< _Tp >](#)
- struct [std::atomic< _Tp * >](#)
- struct [std::atomic< bool >](#)
- struct [std::atomic< char >](#)
- struct [std::atomic< char16_t >](#)
- struct [std::atomic< char32_t >](#)
- struct [std::atomic< int >](#)
- struct [std::atomic< long >](#)
- struct [std::atomic< long long >](#)
- struct [std::atomic< short >](#)
- struct [std::atomic< signed char >](#)
- struct [std::atomic< unsigned char >](#)
- struct [std::atomic< unsigned int >](#)
- struct [std::atomic< unsigned long >](#)
- struct [std::atomic< unsigned long long >](#)
- struct [std::atomic< unsigned short >](#)
- struct [std::atomic< wchar_t >](#)

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_ATOMIC`

Typedefs

- typedef atomic< bool > [std::atomic_bool](#)
- typedef atomic< char > [std::atomic_char](#)
- typedef atomic< char16_t > [std::atomic_char16_t](#)
- typedef atomic< char32_t > [std::atomic_char32_t](#)
- typedef atomic< int > [std::atomic_int](#)
- typedef atomic< int_fast16_t > [std::atomic_int_fast16_t](#)
- typedef atomic< int_fast32_t > [std::atomic_int_fast32_t](#)
- typedef atomic< int_fast64_t > [std::atomic_int_fast64_t](#)
- typedef atomic< int_fast8_t > [std::atomic_int_fast8_t](#)
- typedef atomic< int_least16_t > [std::atomic_int_least16_t](#)
- typedef atomic< int_least32_t > [std::atomic_int_least32_t](#)
- typedef atomic< int_least64_t > [std::atomic_int_least64_t](#)
- typedef atomic< int_least8_t > [std::atomic_int_least8_t](#)
- typedef atomic< intmax_t > [std::atomic_intmax_t](#)
- typedef atomic< intptr_t > [std::atomic_intptr_t](#)
- typedef atomic< long long > [std::atomic_llong](#)
- typedef atomic< long > [std::atomic_long](#)
- typedef atomic< ptrdiff_t > [std::atomic_ptrdiff_t](#)
- typedef atomic< signed char > [std::atomic_schar](#)
- typedef atomic< short > [std::atomic_short](#)
- typedef atomic< size_t > [std::atomic_size_t](#)
- typedef atomic< unsigned char > [std::atomic_uchar](#)
- typedef atomic< unsigned int > [std::atomic_uint](#)
- typedef atomic< uint_fast16_t > [std::atomic_uint_fast16_t](#)
- typedef atomic< uint_fast32_t > [std::atomic_uint_fast32_t](#)
- typedef atomic< uint_fast64_t > [std::atomic_uint_fast64_t](#)
- typedef atomic< uint_fast8_t > [std::atomic_uint_fast8_t](#)
- typedef atomic< uint_least16_t > [std::atomic_uint_least16_t](#)
- typedef atomic< uint_least32_t > [std::atomic_uint_least32_t](#)
- typedef atomic< uint_least64_t > [std::atomic_uint_least64_t](#)
- typedef atomic< uint_least8_t > [std::atomic_uint_least8_t](#)
- typedef atomic< uintmax_t > [std::atomic_uintmax_t](#)
- typedef atomic< uintptr_t > [std::atomic_uintptr_t](#)
- typedef atomic< unsigned long long > [std::atomic_ullong](#)
- typedef atomic< unsigned long > [std::atomic_ulong](#)
- typedef atomic< unsigned short > [std::atomic_ushort](#)
- typedef atomic< wchar_t > [std::atomic_wchar_t](#)

Functions

- template<typename _ITp >
bool **std::atomic_compare_exchange_strong** (atomic< _ITp > *__a, _ITp *__i1, _ITp __i2) noexcept
- template<typename _ITp >
bool **std::atomic_compare_exchange_strong** (volatile atomic< _ITp > *__a, _ITp *__i1, _ITp __i2) noexcept
- template<typename _ITp >
bool **std::atomic_compare_exchange_strong_explicit** (atomic< _ITp > *__a, _ITp *__i1, _ITp __i2, memory_order __m1, memory_order __m2) noexcept

- `template<typename _ITp >`
`bool std::atomic_compare_exchange_strong_explicit (volatile atomic< _ITp > *__a, _ITp *__i1, _ITp __i2,`
`memory_order __m1, memory_order __m2) noexcept`
- `template<typename _ITp >`
`bool std::atomic_compare_exchange_weak (atomic< _ITp > *__a, _ITp *__i1, _ITp __i2) noexcept`
- `template<typename _ITp >`
`bool std::atomic_compare_exchange_weak (volatile atomic< _ITp > *__a, _ITp *__i1, _ITp __i2) noexcept`
- `template<typename _ITp >`
`bool std::atomic_compare_exchange_weak_explicit (atomic< _ITp > *__a, _ITp *__i1, _ITp __i2, memory_order __m1, memory_order __m2) noexcept`
- `template<typename _ITp >`
`bool std::atomic_compare_exchange_weak_explicit (volatile atomic< _ITp > *__a, _ITp *__i1, _ITp __i2,`
`memory_order __m1, memory_order __m2) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_exchange (atomic< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_exchange (volatile atomic< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_exchange_explicit (atomic< _ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_exchange_explicit (volatile atomic< _ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_add (__atomic_base< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_add (volatile __atomic_base< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp * std::atomic_fetch_add (volatile atomic< _ITp * > *__a, ptrdiff_t __d) noexcept`
- `template<typename _ITp >`
`_ITp * std::atomic_fetch_add (atomic< _ITp * > *__a, ptrdiff_t __d) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_add_explicit (__atomic_base< _ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_add_explicit (volatile __atomic_base< _ITp > *__a, _ITp __i, memory_order __m)`
`noexcept`
- `template<typename _ITp >`
`_ITp * std::atomic_fetch_add_explicit (atomic< _ITp * > *__a, ptrdiff_t __d, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp * std::atomic_fetch_add_explicit (volatile atomic< _ITp * > *__a, ptrdiff_t __d, memory_order __m)`
`noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_and (__atomic_base< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_and (volatile __atomic_base< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_and_explicit (__atomic_base< _ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_and_explicit (volatile __atomic_base< _ITp > *__a, _ITp __i, memory_order __m)`
`noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_or (__atomic_base< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_or (volatile __atomic_base< _ITp > *__a, _ITp __i) noexcept`

- `template<typename _ITp >`
`_ITp std::atomic_fetch_or_explicit (__atomic_base< _ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_or_explicit (volatile __atomic_base< _ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_sub (__atomic_base< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_sub (volatile __atomic_base< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp * std::atomic_fetch_sub (volatile atomic< _ITp * > *__a, ptrdiff_t __d) noexcept`
- `template<typename _ITp >`
`_ITp * std::atomic_fetch_sub (atomic< _ITp * > *__a, ptrdiff_t __d) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_sub_explicit (__atomic_base< _ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_sub_explicit (volatile __atomic_base< _ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp * std::atomic_fetch_sub_explicit (volatile atomic< _ITp * > *__a, ptrdiff_t __d, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp * std::atomic_fetch_sub_explicit (atomic< _ITp * > *__a, ptrdiff_t __d, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_xor (__atomic_base< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_xor (volatile __atomic_base< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_xor_explicit (__atomic_base< _ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_fetch_xor_explicit (volatile __atomic_base< _ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `void std::atomic_flag_clear (atomic_flag *__a) noexcept`
- `void std::atomic_flag_clear (volatile atomic_flag *__a) noexcept`
- `void std::atomic_flag_clear_explicit (atomic_flag *__a, memory_order __m) noexcept`
- `void std::atomic_flag_clear_explicit (volatile atomic_flag *__a, memory_order __m) noexcept`
- `bool std::atomic_flag_test_and_set (atomic_flag *__a) noexcept`
- `bool std::atomic_flag_test_and_set (volatile atomic_flag *__a) noexcept`
- `bool std::atomic_flag_test_and_set_explicit (atomic_flag *__a, memory_order __m) noexcept`
- `bool std::atomic_flag_test_and_set_explicit (volatile atomic_flag *__a, memory_order __m) noexcept`
- `template<typename _ITp >`
`void std::atomic_init (atomic< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`void std::atomic_init (volatile atomic< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`bool std::atomic_is_lock_free (const atomic< _ITp > *__a) noexcept`
- `template<typename _ITp >`
`bool std::atomic_is_lock_free (const volatile atomic< _ITp > *__a) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_load (const atomic< _ITp > *__a) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_load (const volatile atomic< _ITp > *__a) noexcept`

- `template<typename _ITp >`
`_ITp std::atomic_load_explicit (const atomic< _ITp > *__a, memory_order __m) noexcept`
- `template<typename _ITp >`
`_ITp std::atomic_load_explicit (const volatile atomic< _ITp > *__a, memory_order __m) noexcept`
- `template<typename _ITp >`
`void std::atomic_store (atomic< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`void std::atomic_store (volatile atomic< _ITp > *__a, _ITp __i) noexcept`
- `template<typename _ITp >`
`void std::atomic_store_explicit (atomic< _ITp > *__a, _ITp __i, memory_order __m) noexcept`
- `template<typename _ITp >`
`void std::atomic_store_explicit (volatile atomic< _ITp > *__a, _ITp __i, memory_order __m) noexcept`

6.21.1 Detailed Description

This is a Standard C++ Library header.

6.22 atomic_base.h File Reference

Classes

- struct [std::__atomic_base< _ITp >](#)
- struct [std::__atomic_base< _ITp >](#)
- struct [std::__atomic_base< _PTp * >](#)
- struct [std::__atomic_flag_base](#)
- struct [std::atomic< _Tp >](#)
- struct [std::atomic_flag](#)

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_ALWAYS_INLINE`
- `#define ATOMIC_FLAG_INIT`
- `#define ATOMIC_VAR_INIT(_VI)`

Typedefs

- typedef unsigned char [std::__atomic_flag_data_type](#)
- typedef enum [std::memory_order](#) [std::memory_order](#)

Enumerations

- enum `__memory_order_modifier` { `__memory_order_mask`, `__memory_order_modifier_mask`, `__memory_order_hle_acquire`, `__memory_order_hle_release` }
- enum `std::memory_order` { `memory_order_relaxed`, `memory_order_consume`, `memory_order_acquire`, `memory_order_release`, `memory_order_acq_rel`, `memory_order_seq_cst` }

Functions

- `std::__attribute__((__always_inline__)) void atomic_thread_fence(memory_order __m) noexcept`
- `constexpr memory_order std::__cmpexch_failure_order(memory_order __m) noexcept`
- `constexpr memory_order std::__cmpexch_failure_order2(memory_order __m) noexcept`
- `template<typename _Tp> _Tp std::kill_dependency(_Tp __y) noexcept`
- `constexpr memory_order std::operator&(memory_order __m, __memory_order_modifier __mod)`
- `constexpr memory_order std::operator|(memory_order __m, __memory_order_modifier __mod)`

6.22.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<atomic>`.

6.23 `atomic_futex.h` File Reference

Namespaces

- `std`

6.23.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly.

6.24 `atomic_lockfree_defines.h` File Reference

Macros

- `#define ATOMIC_BOOL_LOCK_FREE`
- `#define ATOMIC_CHAR16_T_LOCK_FREE`
- `#define ATOMIC_CHAR32_T_LOCK_FREE`
- `#define ATOMIC_CHAR_LOCK_FREE`
- `#define ATOMIC_INT_LOCK_FREE`
- `#define ATOMIC_LLONG_LOCK_FREE`
- `#define ATOMIC_LONG_LOCK_FREE`
- `#define ATOMIC_POINTER_LOCK_FREE`
- `#define ATOMIC_SHORT_LOCK_FREE`
- `#define ATOMIC_WCHAR_T_LOCK_FREE`

6.24.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<atomic>`.

6.25 atomic_word.h File Reference

Macros

- `#define _GLIBCXX_READ_MEM_BARRIER`
- `#define _GLIBCXX_WRITE_MEM_BARRIER`

Typedefs

- `typedef int _Atomic_word`

6.25.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.26 atomicity.h File Reference

Namespaces

- [`__gnu_cxx`](#)

Macros

- `#define _GLIBCXX_READ_MEM_BARRIER`
- `#define _GLIBCXX_WRITE_MEM_BARRIER`

Functions

- `static void __gnu_cxx::__atomic_add_single (_Atomic_word * __mem, int __val)`
- `else __gnu_cxx::__atomic_add_single (__mem, __val)`
- `_Atomic_word __gnu_cxx::__attribute__((__unused__)) __exchange_and_add(volatile _Atomic_word *`
- `static _Atomic_word __gnu_cxx::__exchange_and_add_single (_Atomic_word * __mem, int __val)`
- `else return __gnu_cxx::__exchange_and_add_single (__mem, __val)`
- `_Atomic_word int __gnu_cxx::throw ()`

Variables

- `static _Atomic_word int __gnu_cxx::__val`

6.26.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.27 `auto_ptr.h` File Reference

Classes

- class [std::auto_ptr<_Tp>](#)
- struct [std::auto_ptr_ref<_Tp1>](#)

Namespaces

- [std](#)

6.27.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

6.28 `backward_warning.h` File Reference

6.28.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iosfwd>`.

6.29 `balanced_quicksort.h` File Reference

Classes

- struct [__gnu_parallel::__QSBThreadLocal<_RAIter>](#)

Namespaces

- [__gnu_parallel](#)

Functions

- `template<typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_sort_qsb (_RAIter __begin, _RAIter __end, _Compare __comp, _ThreadIndex __num_threads)`
- `template<typename _RAIter, typename _Compare >`
`void __gnu_parallel::__qsb_conquer (_QSBThreadLocal< _RAIter > *__tls, _RAIter __begin, _RAIter __end, _Compare __comp, _ThreadIndex __iam, _ThreadIndex __num_threads, bool __parent_wait)`
- `template<typename _RAIter, typename _Compare >`
`std::iterator_traits< _RAIter >::difference_type __gnu_parallel::__qsb_divide (_RAIter __begin, _RAIter __end, _Compare __comp, _ThreadIndex __num_threads)`
- `template<typename _RAIter, typename _Compare >`
`void __gnu_parallel::__qsb_local_sort_with_helping (_QSBThreadLocal< _RAIter > *__tls, _Compare &__comp, _ThreadIndex __iam, bool __wait)`

6.29.1 Detailed Description

Implementation of a dynamically load-balanced parallel quicksort.

It works in-place and needs only logarithmic extra memory. The algorithm is similar to the one proposed in

P. Tsigas and Y. Zhang. A simple, fast parallel implementation of quicksort and its performance evaluation on SUN enterprise 10000. In 11th Euromicro Conference on Parallel, Distributed and Network-Based Processing, page 372, 2003.

This file is a GNU parallel extension to the Standard C++ Library.

6.30 base.h File Reference

Namespaces

- [__gnu_profile](#)
- [std](#)
- [std::__profile](#)

6.30.1 Detailed Description

Sequential helper functions. This file is a GNU profile extension to the Standard C++ Library.

6.31 base.h File Reference

Classes

- `class __gnu_parallel::__binder1st< _Operation, _FirstArgumentType, _SecondArgumentType, _ResultType >`
- `class __gnu_parallel::__binder2nd< _Operation, _FirstArgumentType, _SecondArgumentType, _ResultType >`
- `class __gnu_parallel::__unary_negate< _Predicate, argument_type >`
- `class __gnu_parallel::__EqualFromLess< _T1, _T2, _Compare >`
- `struct __gnu_parallel::__EqualTo< _T1, _T2 >`
- `struct __gnu_parallel::__Less< _T1, _T2 >`
- `struct __gnu_parallel::__Multiplies< _Tp1, _Tp2, _Result >`
- `struct __gnu_parallel::__Plus< _Tp1, _Tp2, _Result >`
- `class __gnu_parallel::__PseudoSequence< _Tp, _DifferenceTp >`
- `class __gnu_parallel::__PseudoSequenceIterator< _Tp, _DifferenceTp >`

Namespaces

- [__gnu_parallel](#)
- [__gnu_sequential](#)
- [std](#)
- [std::__parallel](#)

Macros

- `#define _GLIBCXX_PARALLEL_ASSERT(_Condition)`

Functions

- `void __gnu_parallel::__decode2 (_CASable __x, int &__a, int &__b)`
- `_CASable __gnu_parallel::__encode2 (int __a, int __b)`
- `_ThreadIndex __gnu_parallel::__get_max_threads ()`
- `bool __gnu_parallel::__is_parallel (const _Parallelism __p)`
- `template<typename _RAIter, typename _Compare >
_RAIter __gnu_parallel::__median_of_three_iterators (_RAIter __a, _RAIter __b, _RAIter __c, _Compare __↔
comp)`
- `template<typename _Size >
_Size __gnu_parallel::__rd_log2 (_Size __n)`
- `template<typename _Tp >
const _Tp & __gnu_parallel::max (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp >
const _Tp & __gnu_parallel::min (const _Tp &__a, const _Tp &__b)`

6.31.1 Detailed Description

Sequential helper functions. This file is a GNU parallel extension to the Standard C++ Library.

6.32 [basic_file.h](#) File Reference

Namespaces

- [std](#)

6.32.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ios>`.

6.33 `basic_ios.h` File Reference

Classes

- class `std::basic_ios<_CharT, _Traits>`

Namespaces

- `std`

Functions

- template<typename _Facet >
const _Facet & **std::__check_facet** (const _Facet *__f)

6.33.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ios>`.

6.34 `basic_ios.tcc` File Reference

Namespaces

- `std`

Macros

- `#define _BASIC_IOS_TCC`

6.34.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ios>`.

6.35 `basic_iterator.h` File Reference

6.35.1 Detailed Description

Includes the original header files concerned with iterators except for stream iterators. This file is a GNU parallel extension to the Standard C++ Library.

6.36 basic_string.h File Reference

Classes

- class [std::basic_string<_CharT, _Traits, _Alloc>](#)
- struct [std::hash<string>](#)
- struct [std::hash<u16string>](#)
- struct [std::hash<u32string>](#)
- struct [std::hash<wstring>](#)

Namespaces

- [std](#)

Macros

- `#define __cpp_lib_string_udls`

Functions

- `template<typename _CharT, typename _Traits, typename _Alloc>
basic_istream<_CharT, _Traits> & std::getline (basic_istream<_CharT, _Traits> &__is, basic_string<_CharT, _Traits, _Alloc> &__str, _CharT __delim)`
- `template<typename _CharT, typename _Traits, typename _Alloc>
basic_istream<_CharT, _Traits> & std::getline (basic_istream<_CharT, _Traits> &__is, basic_string<_CharT, _Traits, _Alloc> &__str)`
- `template<typename _CharT, typename _Traits, typename _Alloc>
basic_istream<_CharT, _Traits> & std::getline (basic_istream<_CharT, _Traits> &&__is, basic_string<_↵_CharT, _Traits, _Alloc> &__str, _CharT __delim)`
- `template<typename _CharT, typename _Traits, typename _Alloc>
basic_istream<_CharT, _Traits> & std::getline (basic_istream<_CharT, _Traits> &&__is, basic_string<_↵_CharT, _Traits, _Alloc> &__str)`
- `template<>
basic_istream<char> & std::getline (basic_istream<char> &__in, basic_string<char> &__str, char __↵delim)`
- `template<>
basic_istream<wchar_t> & std::getline (basic_istream<wchar_t> &__in, basic_string<wchar_t> &__str, wchar_t __delim)`
- `template<typename _CharT, typename _Traits, typename _Alloc>
bool std::operator!= (const basic_string<_CharT, _Traits, _Alloc> &__lhs, const basic_string<_CharT, _Traits, _Alloc> &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc>
bool std::operator!= (const _CharT *__lhs, const basic_string<_CharT, _Traits, _Alloc> &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc>
bool std::operator!= (const basic_string<_CharT, _Traits, _Alloc> &__lhs, const _CharT *__rhs)`
- `_GLIBCXX_DEFAULT_ABI_TAG basic_string<char> std::literals::string_literals::operator""s (const char *__str, size_t __len)`
- `_GLIBCXX_DEFAULT_ABI_TAG basic_string<wchar_t> std::literals::string_literals::operator""s (const wchar_t *__str, size_t __len)`

- `_GLIBCXX_DEFAULT_ABI_TAG basic_string< char16_t > std::literals::string_literals::operator""s` (const `char16_t *__str`, `size_t __len`)
- `_GLIBCXX_DEFAULT_ABI_TAG basic_string< char32_t > std::literals::string_literals::operator""s` (const `char32_t *__str`, `size_t __len`)
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > std::operator+ (const basic_string< _CharT, _Traits, _Alloc > &__lhs,`
`const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > std::operator+ (const _CharT *__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > std::operator+ (_CharT __lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > std::operator+ (const basic_string< _CharT, _Traits, _Alloc > &__lhs,`
`const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > std::operator+ (const basic_string< _CharT, _Traits, _Alloc > &__lhs,`
`_CharT __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > std::operator+ (basic_string< _CharT, _Traits, _Alloc > &&__lhs, const`
`basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > std::operator+ (const basic_string< _CharT, _Traits, _Alloc > &__lhs,`
`basic_string< _CharT, _Traits, _Alloc > &&__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > std::operator+ (basic_string< _CharT, _Traits, _Alloc > &&__lhs,`
`basic_string< _CharT, _Traits, _Alloc > &&__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > std::operator+ (const _CharT *__lhs, basic_string< _CharT, _Traits,`
`_Alloc > &&__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > std::operator+ (_CharT __lhs, basic_string< _CharT, _Traits, _Alloc >`
`&&__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > std::operator+ (basic_string< _CharT, _Traits, _Alloc > &&__lhs, const`
`_CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_string< _CharT, _Traits, _Alloc > std::operator+ (basic_string< _CharT, _Traits, _Alloc > &&__lhs, _CharT`
`__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool std::operator< (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const basic_string< _CharT, _Traits,`
`_Alloc > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool std::operator< (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool std::operator< (const _CharT *__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, const basic_ostream<`
`_CharT, _Traits, _Alloc > &__str)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool std::operator<= (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const basic_string< _CharT, _Traits,`
`_Alloc > &__rhs) noexcept`

- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool std::operator<= (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool std::operator<= (const _CharT *__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool std::operator== (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs) noexcept`
- `template<typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, bool >::__type std::operator== (const basic_string< _CharT > &__lhs, const basic_string< _CharT > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool std::operator== (const _CharT *__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool std::operator== (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool std::operator> (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool std::operator> (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool std::operator> (const _CharT *__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool std::operator>= (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs) noexcept`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool std::operator>= (const basic_string< _CharT, _Traits, _Alloc > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`bool std::operator>= (const _CharT *__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__is, basic_string< _CharT, _Traits, _Alloc > &__str)`
- `template<>`
`basic_istream< char > & std::operator>> (basic_istream< char > &__is, basic_string< char > &__str)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`void std::swap (basic_string< _CharT, _Traits, _Alloc > &__lhs, basic_string< _CharT, _Traits, _Alloc > &__rhs) noexcept(/*conditional */)`

6.36.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<string>`.

6.37 `basic_string.tcc` File Reference

Namespaces

- [std](#)

Macros

- `#define _BASIC_STRING_TCC`

Functions

- `template<typename _CharT, typename _Traits, typename _Alloc >
basic_istream< _CharT, _Traits > & std::getline (basic_istream< _CharT, _Traits > &__is, basic_string< _CharT, _Traits, _Alloc > &__str, _CharT __delim)`
- `template<typename _CharT, typename _Traits, typename _Alloc >
basic_string< _CharT, _Traits, _Alloc > std::operator+ (const _CharT *__lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >
basic_string< _CharT, _Traits, _Alloc > std::operator+ (_CharT __lhs, const basic_string< _CharT, _Traits, _Alloc > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc >
basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__is, basic_string< _CharT, _Traits, _Alloc > &__str)`

6.37.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<string>`.

6.38 bin_search_tree.hpp File Reference

Namespaces

- [`__gnu_pbds`](#)

Macros

- `#define PB_DS_ASSERT_NODE_CONSISTENT(_Node)`
- `#define PB_DS_BIN_TREE_NAME`
- `#define PB_DS_BIN_TREE_TRAITS_BASE`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_STRUCT_ONLY_ASSERT_VALID(X)`

6.38.1 Detailed Description

Contains an implementation class for binary search tree.

6.39 `binary_heap.hpp` File Reference

Classes

- class [__gnu_pbds::detail::binary_heap](#)< Value_Type, Cmp_Fn, _Alloc >

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_ASSERT_VALID(X)`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_DEBUG_VERIFY(_Cond)`
- `#define PB_DS_ENTRY_CMP_DEC`
- `#define PB_DS_RESIZE_POLICY_DEC`

6.39.1 Detailed Description

Contains an implementation class for a binary heap.

6.40 `binders.h` File Reference

Classes

- class [std::binder1st](#)< _Operation >
- class [std::binder2nd](#)< _Operation >

Namespaces

- [std](#)

Functions

- `template<typename _Operation , typename _Tp >`
`binder1st< _Operation > std::bind1st (const _Operation &__fn, const _Tp &__x)`
- `template<typename _Operation , typename _Tp >`
`binder2nd< _Operation > std::bind2nd (const _Operation &__fn, const _Tp &__x)`

6.40.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<functional>`.

6.41 binomial_heap_.hpp File Reference

Classes

- class [__gnu_pbds::detail::binomial_heap](#)< Value_Type, Cmp_Fn, _Alloc >

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`

6.41.1 Detailed Description

Contains an implementation class for a binomial heap.

6.42 binomial_heap_base_.hpp File Reference

Classes

- class [__gnu_pbds::detail::binomial_heap_base](#)< Value_Type, Cmp_Fn, _Alloc >

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_ASSERT_BASE_NODE_CONSISTENT(_Node, _Bool)`
- `#define PB_DS_ASSERT_VALID_COND(X, _StrictlyBinomial)`
- `#define PB_DS_B_HEAP_BASE`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`

6.42.1 Detailed Description

Contains an implementation class for a base of binomial heaps.

6.43 bitmap_allocator.h File Reference

Classes

- class [__gnu_cxx::__detail::__mini_vector<_Tp>](#)
- class [__gnu_cxx::__detail::__Bitmap_counter<_Tp>](#)
- class [__gnu_cxx::__detail::__Ffit_finder<_Tp>](#)
- class [__gnu_cxx::bitmap_allocator<_Tp>](#)
- class [__gnu_cxx::bitmap_allocator<_Tp>](#)
- class [__gnu_cxx::free_list](#)

Namespaces

- [__gnu_cxx](#)
- [__gnu_cxx::__detail](#)

Macros

- [#define _BALLOC_ALIGN_BYTES](#)

Enumerations

- enum { [bits_per_byte](#), [bits_per_block](#) }

Functions

- void [__gnu_cxx::__detail::__bit_allocate](#) (size_t *__pbmap, size_t __pos) throw ()
- void [__gnu_cxx::__detail::__bit_free](#) (size_t *__pbmap, size_t __pos) throw ()
- template<typename _ForwardIterator, typename _Tp, typename _Compare >
_ForwardIterator [__gnu_cxx::__detail::__lower_bound](#) (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)
- template<typename _AddrPair >
size_t [__gnu_cxx::__detail::__num_bitmaps](#) (_AddrPair __ap)
- template<typename _AddrPair >
size_t [__gnu_cxx::__detail::__num_blocks](#) (_AddrPair __ap)
- size_t [__gnu_cxx::__Bit_scan_forward](#) (size_t __num)
- template<typename _Tp1, typename _Tp2 >
bool [__gnu_cxx::operator!=](#) (const bitmap_allocator<_Tp1> &, const bitmap_allocator<_Tp2> &) throw ()
- template<typename _Tp1, typename _Tp2 >
bool [__gnu_cxx::operator==](#) (const bitmap_allocator<_Tp1> &, const bitmap_allocator<_Tp2> &) throw ()

6.43.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.43.2 Macro Definition Documentation

6.43.2.1 `#define _BALLOC_ALIGN_BYTES`

The constant in the expression below is the alignment required in bytes.

Definition at line 43 of file `bitmap_allocator.h`.

6.44 **bitset** File Reference

Classes

- struct [std::_Base_bitset< _Nw >](#)
- struct [std::_Base_bitset< 0 >](#)
- struct [std::_Base_bitset< 1 >](#)
- class [std::bitset< _Nb >](#)
- class [std::bitset< _Nb >::reference](#)
- struct [std::hash<::bitset< _Nb > >](#)

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_BITSET`
- `#define _GLIBCXX_BITSET_BITS_PER_ULL`
- `#define _GLIBCXX_BITSET_BITS_PER_WORD`
- `#define _GLIBCXX_BITSET_WORDS(__n)`

Functions

- `template<size_t _Nb>`
`bitset< _Nb > std::operator& (const bitset< _Nb > &__x, const bitset< _Nb > &__y) noexcept`
- `template<size_t _Nb>`
`bitset< _Nb > std::operator| (const bitset< _Nb > &__x, const bitset< _Nb > &__y) noexcept`
- `template<size_t _Nb>`
`bitset< _Nb > std::operator^ (const bitset< _Nb > &__x, const bitset< _Nb > &__y) noexcept`
- `template<class _CharT, class _Traits, size_t _Nb>`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is, bitset< _Nb > &__x)`
- `template<class _CharT, class _Traits, size_t _Nb>`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const bitset< _Nb > &__x)`

6.44.1 Detailed Description

This is a Standard C++ Library header.

6.45 `bitset` File Reference

Classes

- class [std::__debug::bitset<_Nb>](#)
- struct [std::hash<__debug::bitset<_Nb>>](#)

Namespaces

- [std](#)
- [std::__debug](#)

Functions

- `template<size_t _Nb>`
`bitset<_Nb> std::__debug::operator& (const bitset<_Nb> &__x, const bitset<_Nb> &__y) noexcept`
- `template<typename _CharT, typename _Traits, size_t _Nb>`
`std::basic_ostream<_CharT, _Traits> & std::__debug::operator<< (std::basic_ostream<_CharT, _Traits> &__os, const bitset<_Nb> &__x)`
- `template<typename _CharT, typename _Traits, size_t _Nb>`
`std::basic_istream<_CharT, _Traits> & std::__debug::operator>> (std::basic_istream<_CharT, _Traits> &__is, bitset<_Nb> &__x)`
- `template<size_t _Nb>`
`bitset<_Nb> std::__debug::operator^ (const bitset<_Nb> &__x, const bitset<_Nb> &__y) noexcept`
- `template<size_t _Nb>`
`bitset<_Nb> std::__debug::operator| (const bitset<_Nb> &__x, const bitset<_Nb> &__y) noexcept`

6.45.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.46 `bitset` File Reference

Classes

- class [std::__profile::bitset<_Nb>](#)
- struct [std::hash<__profile::bitset<_Nb>>](#)

Namespaces

- [std](#)
- [std::__profile](#)

Functions

- `template<size_t _Nb>
bitset< _Nb > std::__profile::operator& (const bitset< _Nb > &__x, const bitset< _Nb > &__y) noexcept`
- `template<typename _CharT, typename _Traits, size_t _Nb>
std::basic_ostream< _CharT, _Traits > & std::__profile::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const bitset< _Nb > &__x)`
- `template<typename _CharT, typename _Traits, size_t _Nb>
std::basic_istream< _CharT, _Traits > & std::__profile::operator>> (std::basic_istream< _CharT, _Traits > &__is, bitset< _Nb > &__x)`
- `template<size_t _Nb>
bitset< _Nb > std::__profile::operator^ (const bitset< _Nb > &__x, const bitset< _Nb > &__y) noexcept`
- `template<size_t _Nb>
bitset< _Nb > std::__profile::operator| (const bitset< _Nb > &__x, const bitset< _Nb > &__y) noexcept`

6.46.1 Detailed Description

This file is a GNU profile extension to the Standard C++ Library.

6.47 bool_set File Reference

Classes

- class [std::tr2::bool_set](#)

Namespaces

- [std](#)
- [std::tr2](#)

Macros

- `#define _GLIBCXX_TR2_BOOL_SET`

Functions

- `bool std::tr2::certainly (bool_set __b)`
- `bool std::tr2::contains (bool_set __s, bool_set __t)`
- `bool std::tr2::equals (bool_set __s, bool_set __t)`
- `bool std::tr2::is_emptyset (bool_set __b)`
- `bool std::tr2::is_indeterminate (bool_set __b)`
- `bool std::tr2::is_singleton (bool_set __b)`
- `bool_set std::tr2::operator!= (bool __s, bool_set __t)`
- `bool_set std::tr2::operator!= (bool_set __s, bool __t)`
- `bool_set std::tr2::operator!= (bool_set __s, bool_set __t)`
- `bool_set std::tr2::operator& (bool __s, bool_set __t)`
- `bool_set std::tr2::operator& (bool_set __s, bool __t)`
- `bool_set std::tr2::operator== (bool __s, bool_set __t)`
- `bool_set std::tr2::operator== (bool_set __s, bool __t)`
- `bool_set std::tr2::operator^ (bool __s, bool_set __t)`
- `bool_set std::tr2::operator^ (bool_set __s, bool __t)`
- `bool_set std::tr2::operator| (bool __s, bool_set __t)`
- `bool_set std::tr2::operator| (bool_set __s, bool __t)`
- `bool std::tr2::possibly (bool_set __b)`
- `bool_set std::tr2::set_complement (bool_set __b)`
- `bool_set std::tr2::set_intersection (bool __s, bool_set __t)`
- `bool_set std::tr2::set_intersection (bool_set __s, bool __t)`
- `bool_set std::tr2::set_intersection (bool_set __s, bool_set __t)`
- `bool_set std::tr2::set_union (bool __s, bool_set __t)`
- `bool_set std::tr2::set_union (bool_set __s, bool __t)`
- `bool_set std::tr2::set_union (bool_set __s, bool_set __t)`

6.47.1 Detailed Description

This is a TR2 C++ Library header.

6.48 `bool_set.tcc` File Reference

Namespaces

- [std](#)
- [std::tr2](#)

Macros

- `#define _GLIBCXX_TR2_BOOL_SET_TCC`

6.48.1 Detailed Description

This is a TR2 C++ Library header.

6.49 boost_concept_check.h File Reference

Namespaces

- [__gnu_cxx](#)

Macros

- `#define _GLIBCXX_CLASS_REQUIRES(_type_var, _ns, _concept)`
- `#define _GLIBCXX_CLASS_REQUIRES2(_type_var1, _type_var2, _ns, _concept)`
- `#define _GLIBCXX_CLASS_REQUIRES3(_type_var1, _type_var2, _type_var3, _ns, _concept)`
- `#define _GLIBCXX_CLASS_REQUIRES4(_type_var1, _type_var2, _type_var3, _type_var4, _ns, _concept)`
- `#define _GLIBCXX_DEFINE_BINARY_OPERATOR_CONSTRAINT(_OP, _NAME)`
- `#define _GLIBCXX_DEFINE_BINARY_PREDICATE_OP_CONSTRAINT(_OP, _NAME)`
- `#define _IsUnused`

Functions

- `template<class _Tp >`
`void __gnu_cxx::__aux_require_boolean_expr (const _Tp &__t)`
- `void __gnu_cxx::__error_type_must_be_a_signed_integer_type ()`
- `void __gnu_cxx::__error_type_must_be_an_integer_type ()`
- `void __gnu_cxx::__error_type_must_be_an_unsigned_integer_type ()`
- `template<class _Concept >`
`void __gnu_cxx::__function_requires ()`

6.49.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iterator>`.

6.50 branch_policy.hpp File Reference

Classes

- `struct __gnu_pbds::detail::branch_policy< Node_Cltr, Node_Itr, _Alloc >`
- `struct __gnu_pbds::detail::branch_policy< Node_Cltr, Node_Cltr, _Alloc >`

Namespaces

- [__gnu_pbds](#)

6.50.1 Detailed Description

Contains a base class for branch policies.

6.51 `c++0x_warning.h` File Reference

6.51.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iosfwd>`.

6.52 `c++14_warning.h` File Reference

6.52.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iosfwd>`.

6.53 `c++allocator.h` File Reference

Namespaces

- [std](#)

Typedefs

- `template<typename _Tp >`
`using std::__allocator_base = __gnu_cxx::new_allocator< _Tp >`

6.53.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

6.54 `c++config.h` File Reference

Namespaces

- [__gnu_cxx](#)
- [std](#)

Macros

- `#define __GLIBCXX__`
- `#define __glibcxx_assert(_Condition)`
- `#define __N(msgid)`
- `#define _GLIBCXX11_USE_C99_COMPLEX`
- `#define _GLIBCXX11_USE_C99_MATH`
- `#define _GLIBCXX11_USE_C99_STDIO`
- `#define _GLIBCXX11_USE_C99_STDLIB`
- `#define _GLIBCXX11_USE_C99_WCHAR`
- `#define _GLIBCXX14_CONSTEXPR`
- `#define _GLIBCXX98_USE_C99_COMPLEX`
- `#define _GLIBCXX98_USE_C99_MATH`
- `#define _GLIBCXX98_USE_C99_STDIO`
- `#define _GLIBCXX98_USE_C99_STDLIB`
- `#define _GLIBCXX98_USE_C99_WCHAR`
- `#define _GLIBCXX_ABI_TAG_CXX11`
- `#define _GLIBCXX_ATOMIC_BUILTINS`
- `#define _GLIBCXX_BEGIN_EXTERN_C`
- `#define _GLIBCXX_BEGIN_NAMESPACE_CXX11`
- `#define _GLIBCXX_BEGIN_NAMESPACE_LDBL`
- `#define _GLIBCXX_BEGIN_NAMESPACE_LDBL_OR_CXX11`
- `#define _GLIBCXX_BEGIN_NAMESPACE_VERSION`
- `#define _GLIBCXX_DEFAULT_ABI_TAG`
- `#define _GLIBCXX_DEPRECATED`
- `#define _GLIBCXX_END_EXTERN_C`
- `#define _GLIBCXX_END_NAMESPACE_CXX11`
- `#define _GLIBCXX_END_NAMESPACE_LDBL`
- `#define _GLIBCXX_END_NAMESPACE_LDBL_OR_CXX11`
- `#define _GLIBCXX_END_NAMESPACE_VERSION`
- `#define _GLIBCXX_EXTERN_TEMPLATE`
- `#define _GLIBCXX_FAST_MATH`
- `#define _GLIBCXX_FULLY_DYNAMIC_STRING`
- `#define _GLIBCXX_HAVE__CXA_THREAD_ATEXIT_IMPL`
- `#define _GLIBCXX_HAVE_ACOSF`
- `#define _GLIBCXX_HAVE_ACOSL`
- `#define _GLIBCXX_HAVE_AS_SYMVER_DIRECTIVE`
- `#define _GLIBCXX_HAVE_ASINF`
- `#define _GLIBCXX_HAVE_ASINL`
- `#define _GLIBCXX_HAVE_AT_QUICK_EXIT`
- `#define _GLIBCXX_HAVE_ATAN2F`
- `#define _GLIBCXX_HAVE_ATAN2L`
- `#define _GLIBCXX_HAVE_ATANF`
- `#define _GLIBCXX_HAVE_ATANL`
- `#define _GLIBCXX_HAVE_ATTRIBUTE_VISIBILITY`
- `#define _GLIBCXX_HAVE_CEILF`
- `#define _GLIBCXX_HAVE_CEILL`
- `#define _GLIBCXX_HAVE_COMPLEX_H`
- `#define _GLIBCXX_HAVE_COSF`
- `#define _GLIBCXX_HAVE_COSHF`
- `#define _GLIBCXX_HAVE_COSHL`

- `#define _GLIBCXX_HAVE_COSL`
- `#define _GLIBCXX_HAVE_DIRENT_H`
- `#define _GLIBCXX_HAVE_DLFCN_H`
- `#define _GLIBCXX_HAVE_EBADMSG`
- `#define _GLIBCXX_HAVE_ECANCELED`
- `#define _GLIBCXX_HAVE_ECHILD`
- `#define _GLIBCXX_HAVE_EIDRM`
- `#define _GLIBCXX_HAVE_ENDIAN_H`
- `#define _GLIBCXX_HAVE_ENODATA`
- `#define _GLIBCXX_HAVE_ENOLINK`
- `#define _GLIBCXX_HAVE_ENOSPC`
- `#define _GLIBCXX_HAVE_ENOSR`
- `#define _GLIBCXX_HAVE_ENOSTR`
- `#define _GLIBCXX_HAVE_ENOTRECOVERABLE`
- `#define _GLIBCXX_HAVE_ENOTSUP`
- `#define _GLIBCXX_HAVE_EOVERFLOW`
- `#define _GLIBCXX_HAVE_EOWNERDEAD`
- `#define _GLIBCXX_HAVE_EPERM`
- `#define _GLIBCXX_HAVE_EPROTO`
- `#define _GLIBCXX_HAVE_ETIME`
- `#define _GLIBCXX_HAVE_ETIMEDOUT`
- `#define _GLIBCXX_HAVE_ETXTBSY`
- `#define _GLIBCXX_HAVE_EWOULDBLOCK`
- `#define _GLIBCXX_HAVE_EXECINFO_H`
- `#define _GLIBCXX_HAVE_EXPF`
- `#define _GLIBCXX_HAVE_EXPL`
- `#define _GLIBCXX_HAVE_FABSF`
- `#define _GLIBCXX_HAVE_FABSL`
- `#define _GLIBCXX_HAVE_FCNTL_H`
- `#define _GLIBCXX_HAVE_FENV_H`
- `#define _GLIBCXX_HAVE_FINITE`
- `#define _GLIBCXX_HAVE_FINITEF`
- `#define _GLIBCXX_HAVE_FINITEL`
- `#define _GLIBCXX_HAVE_FLOAT_H`
- `#define _GLIBCXX_HAVE_FLOORF`
- `#define _GLIBCXX_HAVE_FLOORL`
- `#define _GLIBCXX_HAVE_FMODF`
- `#define _GLIBCXX_HAVE_FMODL`
- `#define _GLIBCXX_HAVE_FREXPF`
- `#define _GLIBCXX_HAVE_FREXPL`
- `#define _GLIBCXX_HAVE_GETIPINFO`
- `#define _GLIBCXX_HAVE_GETS`
- `#define _GLIBCXX_HAVE_HYPOT`
- `#define _GLIBCXX_HAVE_HYPOTF`
- `#define _GLIBCXX_HAVE_HYPOTL`
- `#define _GLIBCXX_HAVE_ICONV`
- `#define _GLIBCXX_HAVE_INT64_T`
- `#define _GLIBCXX_HAVE_INT64_T_LONG`
- `#define _GLIBCXX_HAVE_INTTYPES_H`
- `#define _GLIBCXX_HAVE_ISINF`
- `#define _GLIBCXX_HAVE_ISINFF`

- `#define _GLIBCXX_HAVE_ISINFL`
- `#define _GLIBCXX_HAVE_ISNAN`
- `#define _GLIBCXX_HAVE_ISNANF`
- `#define _GLIBCXX_HAVE_ISNANL`
- `#define _GLIBCXX_HAVE_ISWBLANK`
- `#define _GLIBCXX_HAVE_LC_MESSAGES`
- `#define _GLIBCXX_HAVE_LDEXPF`
- `#define _GLIBCXX_HAVE_LDEXPL`
- `#define _GLIBCXX_HAVE_LIBINTL_H`
- `#define _GLIBCXX_HAVE_LIMIT_AS`
- `#define _GLIBCXX_HAVE_LIMIT_DATA`
- `#define _GLIBCXX_HAVE_LIMIT_FSIZE`
- `#define _GLIBCXX_HAVE_LIMIT_RSS`
- `#define _GLIBCXX_HAVE_LIMIT_VMEM`
- `#define _GLIBCXX_HAVE_LINUX_FUTEX`
- `#define _GLIBCXX_HAVE_LOCALE_H`
- `#define _GLIBCXX_HAVE_LOG10F`
- `#define _GLIBCXX_HAVE_LOG10L`
- `#define _GLIBCXX_HAVE_LOGF`
- `#define _GLIBCXX_HAVE_LOGL`
- `#define _GLIBCXX_HAVE_MBSTATE_T`
- `#define _GLIBCXX_HAVE_MEMORY_H`
- `#define _GLIBCXX_HAVE_MODFF`
- `#define _GLIBCXX_HAVE_MODFL`
- `#define _GLIBCXX_HAVE_POLL`
- `#define _GLIBCXX_HAVE_POWF`
- `#define _GLIBCXX_HAVE_POWL`
- `#define _GLIBCXX_HAVE_QUICK_EXIT`
- `#define _GLIBCXX_HAVE_S_ISREG`
- `#define _GLIBCXX_HAVE_SETENV`
- `#define _GLIBCXX_HAVE_SINCOS`
- `#define _GLIBCXX_HAVE_SINCOSF`
- `#define _GLIBCXX_HAVE_SINCOSL`
- `#define _GLIBCXX_HAVE_SINF`
- `#define _GLIBCXX_HAVE_SINHF`
- `#define _GLIBCXX_HAVE_SINHL`
- `#define _GLIBCXX_HAVE_SINL`
- `#define _GLIBCXX_HAVE_SQRTF`
- `#define _GLIBCXX_HAVE_SQRTL`
- `#define _GLIBCXX_HAVE_STDALIGN_H`
- `#define _GLIBCXX_HAVE_STDBOOL_H`
- `#define _GLIBCXX_HAVE_STDINT_H`
- `#define _GLIBCXX_HAVE_STDLIB_H`
- `#define _GLIBCXX_HAVE_STRERROR_L`
- `#define _GLIBCXX_HAVE_STRERROR_R`
- `#define _GLIBCXX_HAVE_STRING_H`
- `#define _GLIBCXX_HAVE_STRINGS_H`
- `#define _GLIBCXX_HAVE_STRTOF`
- `#define _GLIBCXX_HAVE_STRTOLD`
- `#define _GLIBCXX_HAVE_STRUCT_DIRENT_D_TYPE`
- `#define _GLIBCXX_HAVE_STRXFRM_L`

- `#define _GLIBCXX_HAVE_SYMVER_SYMBOL_RENAMING_RUNTIME_SUPPORT`
- `#define _GLIBCXX_HAVE_SYS_IOCTL_H`
- `#define _GLIBCXX_HAVE_SYS_IPC_H`
- `#define _GLIBCXX_HAVE_SYS_PARAM_H`
- `#define _GLIBCXX_HAVE_SYS_RESOURCE_H`
- `#define _GLIBCXX_HAVE_SYS_SDT_H`
- `#define _GLIBCXX_HAVE_SYS_SEM_H`
- `#define _GLIBCXX_HAVE_SYS_STAT_H`
- `#define _GLIBCXX_HAVE_SYS_STATVFS_H`
- `#define _GLIBCXX_HAVE_SYS_SYSINFO_H`
- `#define _GLIBCXX_HAVE_SYS_TIME_H`
- `#define _GLIBCXX_HAVE_SYS_TYPES_H`
- `#define _GLIBCXX_HAVE_SYS_UIO_H`
- `#define _GLIBCXX_HAVE_TANF`
- `#define _GLIBCXX_HAVE_TANHF`
- `#define _GLIBCXX_HAVE_TANHL`
- `#define _GLIBCXX_HAVE_TANL`
- `#define _GLIBCXX_HAVE_TGMATH_H`
- `#define _GLIBCXX_HAVE_TLS`
- `#define _GLIBCXX_HAVE_UCHAR_H`
- `#define _GLIBCXX_HAVE_UNISTD_H`
- `#define _GLIBCXX_HAVE_UTIME_H`
- `#define _GLIBCXX_HAVE_VFWSCANF`
- `#define _GLIBCXX_HAVE_VSWSCANF`
- `#define _GLIBCXX_HAVE_VWSCANF`
- `#define _GLIBCXX_HAVE_WCHAR_H`
- `#define _GLIBCXX_HAVE_WCSTOF`
- `#define _GLIBCXX_HAVE_WCTYPE_H`
- `#define _GLIBCXX_HAVE_WRITEV`
- `#define _GLIBCXX_HOSTED`
- `#define _GLIBCXX_ICONV_CONST`
- `#define _GLIBCXX_INLINE_VERSION`
- `#define _GLIBCXX_MANGLE_SIZE_T`
- `#define _GLIBCXX_NAMESPACE_CXX11`
- `#define _GLIBCXX_NAMESPACE_LDBL`
- `#define _GLIBCXX_NAMESPACE_LDBL_OR_CXX11`
- `#define _GLIBCXX_PACKAGE_GLIBCXX_VERSION`
- `#define _GLIBCXX_PACKAGE_BUGREPORT`
- `#define _GLIBCXX_PACKAGE_NAME`
- `#define _GLIBCXX_PACKAGE_STRING`
- `#define _GLIBCXX_PACKAGE_TARNAME`
- `#define _GLIBCXX_PACKAGE_URL`
- `#define _GLIBCXX_PSEUDO_VISIBILITY(V)`
- `#define _GLIBCXX_RES_LIMITS`
- `#define _GLIBCXX_STDIO_EOF`
- `#define _GLIBCXX_STDIO_SEEK_CUR`
- `#define _GLIBCXX_STDIO_SEEK_END`
- `#define _GLIBCXX_SYMVER`
- `#define _GLIBCXX_SYMVER_GNU`
- `#define _GLIBCXX_SYNCHRONIZATION_HAPPENS_AFTER(A)`
- `#define _GLIBCXX_SYNCHRONIZATION_HAPPENS_BEFORE(A)`

- #define _GLIBCXX_THROW_OR_ABORT(_EXC)
- #define _GLIBCXX_TXN_SAFE
- #define _GLIBCXX_TXN_SAFE_DYN
- #define _GLIBCXX_USE_ALLOCATOR_NEW
- #define _GLIBCXX_USE_C11_UCHAR_CXX11
- #define _GLIBCXX_USE_C99
- #define _GLIBCXX_USE_C99_COMPLEX
- #define _GLIBCXX_USE_C99_COMPLEX_TR1
- #define _GLIBCXX_USE_C99_CTYPE_TR1
- #define _GLIBCXX_USE_C99_FENV_TR1
- #define _GLIBCXX_USE_C99_INTTYPES_TR1
- #define _GLIBCXX_USE_C99_INTTYPES_WCHAR_T_TR1
- #define _GLIBCXX_USE_C99_MATH
- #define _GLIBCXX_USE_C99_MATH_TR1
- #define _GLIBCXX_USE_C99_STDINT_TR1
- #define _GLIBCXX_USE_C99_STDIO
- #define _GLIBCXX_USE_C99_STDLIB
- #define _GLIBCXX_USE_C99_WCHAR
- #define _GLIBCXX_USE_CLOCK_MONOTONIC
- #define _GLIBCXX_USE_CLOCK_REALTIME
- #define _GLIBCXX_USE_CXX11_ABI
- #define _GLIBCXX_USE_DECIMAL_FLOAT
- #define _GLIBCXX_USE_DEPRECATED
- #define _GLIBCXX_USE_DUAL_ABI
- #define _GLIBCXX_USE_FCHMOD
- #define _GLIBCXX_USE_FCHMODAT
- #define _GLIBCXX_USE_FLOAT128
- #define _GLIBCXX_USE_GET_NPROCS
- #define _GLIBCXX_USE_GETTIMEOFDAY
- #define _GLIBCXX_USE_INT128
- #define _GLIBCXX_USE_LFS
- #define _GLIBCXX_USE_LONG_LONG
- #define _GLIBCXX_USE_NANOSLEEP
- #define _GLIBCXX_USE_NLS
- #define _GLIBCXX_USE_RANDOM_TR1
- #define _GLIBCXX_USE_REALPATH
- #define _GLIBCXX_USE_SC_NPROCESSORS_ONLN
- #define _GLIBCXX_USE_SCHED_YIELD
- #define _GLIBCXX_USE_SENDFILE
- #define _GLIBCXX_USE_ST_MTIM
- #define _GLIBCXX_USE_TMPNAM
- #define _GLIBCXX_USE_UTIMENSAT
- #define _GLIBCXX_USE_WCHAR_T
- #define _GLIBCXX_USE_WEAK_REF
- #define _GLIBCXX_VERBOSE
- #define _GLIBCXX_VISIBILITY(V)
- #define _GLIBCXX_WEAK_DEFINITION
- #define _GLIBCXX_X86_RDRAND
- #define _GTHREAD_USE_MUTEX_TIMEDLOCK
- #define LT_OBJDIR
- #define STDC_HEADERS

Typedefs

- typedef __PTRDIFF_TYPE__ **std::ptrdiff_t**
- typedef __SIZE_TYPE__ **std::size_t**

Functions

- namespace __cxx11 **std::__attribute__** ((__abi_tag__("cxx11")))
- namespace __cxx11 **__gnu_cxx::__attribute__** ((__abi_tag__("cxx11")))

Variables

- decltype(nullptr) typedef **std::nullptr_t**

6.54.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iosfwd>`.

6.55 c++io.h File Reference

Namespaces

- [std](#)

Typedefs

- typedef FILE **std::__c_file**
- typedef __pthread_mutex_t **std::__c_lock**

6.55.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ios>`.

6.56 c++locale.h File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_C_LOCALE_GNU`
- `#define _GLIBCXX_NUM_CATEGORIES`

Typedefs

- `typedef __locale_t std::__c_locale`

Functions

- `int std::__convert_from_v (const __c_locale &__cloc __attribute__((__unused__)), char *__out, const int __size __attribute__((__unused__)), const char *__fmt,...)`

6.56.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

6.57 c++locale_internal.h File Reference

Namespaces

- [std](#)

Functions

- Catalogs & `std::get_catalogs ()`

6.57.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

6.58 cassert File Reference

6.58.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `assert.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

6.59 `cast.h` File Reference

Classes

- struct [__gnu_cxx::_Caster<_ToType>](#)

Namespaces

- [__gnu_cxx](#)

Functions

- template<typename _ToType, typename _FromType>
_ToType [__gnu_cxx::const_pointer_cast](#) (const _FromType &__arg)
- template<typename _ToType, typename _FromType>
_ToType [__gnu_cxx::const_pointer_cast](#) (_FromType *__arg)
- template<typename _ToType, typename _FromType>
_ToType [__gnu_cxx::dynamic_pointer_cast](#) (const _FromType &__arg)
- template<typename _ToType, typename _FromType>
_ToType [__gnu_cxx::dynamic_pointer_cast](#) (_FromType *__arg)
- template<typename _ToType, typename _FromType>
_ToType [__gnu_cxx::reinterpret_pointer_cast](#) (const _FromType &__arg)
- template<typename _ToType, typename _FromType>
_ToType [__gnu_cxx::reinterpret_pointer_cast](#) (_FromType *__arg)
- template<typename _ToType, typename _FromType>
_ToType [__gnu_cxx::static_pointer_cast](#) (const _FromType &__arg)
- template<typename _ToType, typename _FromType>
_ToType [__gnu_cxx::static_pointer_cast](#) (_FromType *__arg)

6.59.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ext/pointer.h>`.

6.60 `cc_hash_max_collision_check_resize_trigger_imp.hpp` File Reference

6.60.1 Detailed Description

Contains a resize trigger implementation.

6.61 `cc_ht_map_.hpp` File Reference

Classes

- class [__gnu_pbds::detail::cc_ht_map<Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_Fn, Resize_Policy>](#)

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_CC_HASH_NAME`
- `#define PB_DS_CC_HASH_TRAITS_BASE`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_GEN_POS`
- `#define PB_DS_HASH_EQ_FN_C_DEC`
- `#define PB_DS_RANGED_HASH_FN_C_DEC`

6.61.1 Detailed Description

Contains an implementation class for `cc_ht_map_`.

6.62 ccomplex File Reference

Macros

- `#define _GLIBCXX_CCOMPLEX`

6.62.1 Detailed Description

This is a Standard C++ Library header.

6.63 ccomplex File Reference

Macros

- `#define _GLIBCXX_TR1_CCOMPLEX`

6.63.1 Detailed Description

This is a TR1 C++ Library header.

6.64 cctype File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_CCTYPE`

6.64.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `cctype.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

6.65 cctype File Reference

Macros

- `#define _GLIBCXX_TR1_CCTYPE`

6.65.1 Detailed Description

This is a TR1 C++ Library header.

6.66 cerrno File Reference

Macros

- `#define _GLIBCXX_CERRNO`
- `#define errno`

6.66.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `errno.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

6.67 cfenv File Reference

Macros

- `#define _GLIBCXX_CFENV`

6.67.1 Detailed Description

This is a Standard C++ Library header.

6.68 **cfenv File Reference**

Macros

- `#define _GLIBCXX_TR1_CFENV`

6.68.1 Detailed Description

This is a TR1 C++ Library header.

6.69 **cfloat File Reference**

Macros

- `#define _GLIBCXX_CFLOAT`
- `#define DECIMAL_DIG`
- `#define FLT_EVAL_METHOD`

6.69.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `float.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

6.70 **cfloat File Reference**

Macros

- `#define _GLIBCXX_TR1_CFLOAT`

6.70.1 Detailed Description

This is a TR1 C++ Library header.

6.71 char_traits.h File Reference

Classes

- struct [__gnu_cxx::_Char_types<_CharT>](#)
- struct [__gnu_cxx::char_traits<_CharT>](#)
- struct [std::char_traits<_CharT>](#)
- struct [std::char_traits<char>](#)
- struct [std::char_traits<wchar_t>](#)

Namespaces

- [__gnu_cxx](#)
- [std](#)

6.71.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<string>`.

6.72 checkers.h File Reference

Namespaces

- [__gnu_parallel](#)

Functions

- template<typename _Iter, typename _Compare>
bool [__gnu_parallel::__is_sorted](#) (_Iter __begin, _Iter __end, _Compare __comp)

6.72.1 Detailed Description

Routines for checking the correctness of algorithm results. This file is a GNU parallel extension to the Standard C++ Library.

6.73 chrono File Reference

Classes

- struct [std::chrono::_V2::steady_clock](#)
- struct [std::chrono::_V2::system_clock](#)
- struct [std::chrono::duration<_Rep, _Period>](#)
- struct [std::chrono::duration<_Rep, _Period>](#)
- struct [std::chrono::duration_values<_Rep>](#)
- struct [std::chrono::time_point<_Clock, _Dur>](#)
- struct [std::chrono::time_point<_Clock, _Dur>](#)
- struct [std::chrono::treat_as_floating_point<_Rep>](#)

Namespaces

- [std](#)
- [std::chrono](#)

Macros

- `#define __cpp_lib_chrono_udls`
- `#define _GLIBCXX_CHRONO`

Typedefs

- using [std::chrono::_V2::high_resolution_clock](#) = `system_clock`
- typedef duration< int64_t, ratio< 3600 > > [std::chrono::hours](#)
- typedef duration< int64_t, micro > [std::chrono::microseconds](#)
- typedef duration< int64_t, milli > [std::chrono::milliseconds](#)
- typedef duration< int64_t, ratio< 60 > > [std::chrono::minutes](#)
- typedef duration< int64_t, nano > [std::chrono::nanoseconds](#)
- typedef duration< int64_t > [std::chrono::seconds](#)

Functions

- template<typename _Dur, char... _Digits>
constexpr _Dur [std::literals::chrono_literals::__check_overflow](#) ()
- template<typename _ToDur, typename _Rep, typename _Period >
constexpr enable_if< __is_duration< _ToDur >::value, _ToDur >::type [std::chrono::duration_cast](#) (const duration< _Rep, _Period > &__d)
- template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >
constexpr bool [std::chrono::operator!=](#) (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)
- template<typename _Clock, typename _Dur1, typename _Dur2 >
constexpr bool [std::chrono::operator!=](#) (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 > &__rhs)
- constexpr chrono::duration< long double, ratio< 3600, 1 > > [std::literals::chrono_literals::operator""h](#) (long double __hours)
- template<char... _Digits>
constexpr chrono::hours [std::literals::chrono_literals::operator""h](#) ()
- constexpr chrono::duration< long double, ratio< 60, 1 > > [std::literals::chrono_literals::operator""min](#) (long double __mins)
- template<char... _Digits>
constexpr chrono::minutes [std::literals::chrono_literals::operator""min](#) ()
- constexpr chrono::duration< long double, milli > [std::literals::chrono_literals::operator""ms](#) (long double __msecs)
- template<char... _Digits>
constexpr chrono::milliseconds [std::literals::chrono_literals::operator""ms](#) ()
- constexpr chrono::duration< long double, nano > [std::literals::chrono_literals::operator""ns](#) (long double __nsecs)
- template<char... _Digits>
constexpr chrono::nanoseconds [std::literals::chrono_literals::operator""ns](#) ()

- constexpr chrono::duration< long double > **std::literals::chrono_literals::operator""s** (long double __secs)
- template<char... _Digits>
constexpr chrono::seconds **std::literals::chrono_literals::operator""s** ()
- constexpr chrono::duration< long double, micro > **std::literals::chrono_literals::operator""us** (long double __usecs)
- template<char... _Digits>
constexpr chrono::microseconds **std::literals::chrono_literals::operator""us** ()
- template<typename _Rep1, typename _Period, typename _Rep2 >
constexpr duration< typename __common_rep_type< _Rep1, typename enable_if<!__is_duration< _Rep2 >::value, _Rep2 >::type >::type, _Period > **std::chrono::operator%** (const duration< _Rep1, _Period > &__d, const _Rep2 &__s)
- template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >
constexpr common_type< duration< _Rep1, _Period1 >, duration< _Rep2, _Period2 > >::type **std::chrono::operator%** (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)
- template<typename _Rep1, typename _Period, typename _Rep2 >
constexpr duration< typename __common_rep_type< _Rep1, _Rep2 >::type, _Period > **std::chrono::operator*** (const duration< _Rep1, _Period > &__d, const _Rep2 &__s)
- template<typename _Rep1, typename _Rep2, typename _Period >
constexpr duration< typename __common_rep_type< _Rep2, _Rep1 >::type, _Period > **std::chrono::operator*** (const _Rep1 &__s, const duration< _Rep2, _Period > &__d)
- template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >
constexpr common_type< duration< _Rep1, _Period1 >, duration< _Rep2, _Period2 > >::type **std::chrono::operator+** (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)
- template<typename _Clock, typename _Dur1, typename _Rep2, typename _Period2 >
constexpr time_point< _Clock, typename common_type< _Dur1, duration< _Rep2, _Period2 > >::type > **std::chrono::operator+** (const time_point< _Clock, _Dur1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)
- template<typename _Rep1, typename _Period1, typename _Clock, typename _Dur2 >
constexpr time_point< _Clock, typename common_type< duration< _Rep1, _Period1 >, _Dur2 >::type > **std::chrono::operator+** (const duration< _Rep1, _Period1 > &__lhs, const time_point< _Clock, _Dur2 > &__rhs)
- template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >
constexpr common_type< duration< _Rep1, _Period1 >, duration< _Rep2, _Period2 > >::type **std::chrono::operator-** (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)
- template<typename _Clock, typename _Dur1, typename _Rep2, typename _Period2 >
constexpr time_point< _Clock, typename common_type< _Dur1, duration< _Rep2, _Period2 > >::type > **std::chrono::operator-** (const time_point< _Clock, _Dur1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)
- template<typename _Clock, typename _Dur1, typename _Dur2 >
constexpr common_type< _Dur1, _Dur2 >::type **std::chrono::operator-** (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 > &__rhs)
- template<typename _Rep1, typename _Period, typename _Rep2 >
constexpr duration< typename __common_rep_type< _Rep1, typename enable_if<!__is_duration< _Rep2 >::value, _Rep2 >::type >::type, _Period > **std::chrono::operator/** (const duration< _Rep1, _Period > &__d, const _Rep2 &__s)
- template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >
constexpr common_type< _Rep1, _Rep2 >::type **std::chrono::operator/** (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)
- template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >
constexpr bool **std::chrono::operator<** (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)
- template<typename _Clock, typename _Dur1, typename _Dur2 >
constexpr bool **std::chrono::operator<** (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 > &__rhs)
- template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >
constexpr bool **std::chrono::operator<=** (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)

- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool std::chrono::operator<= (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool std::chrono::operator== (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool std::chrono::operator== (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool std::chrono::operator> (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool std::chrono::operator> (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 > &__rhs)`
- `template<typename _Rep1, typename _Period1, typename _Rep2, typename _Period2 >`
`constexpr bool std::chrono::operator>= (const duration< _Rep1, _Period1 > &__lhs, const duration< _Rep2, _Period2 > &__rhs)`
- `template<typename _Clock, typename _Dur1, typename _Dur2 >`
`constexpr bool std::chrono::operator>= (const time_point< _Clock, _Dur1 > &__lhs, const time_point< _Clock, _Dur2 > &__rhs)`
- `template<typename _ToDur, typename _Clock, typename _Dur >`
`constexpr enable_if< __is_duration< _ToDur >::value, time_point< _Clock, _ToDur > >::type std::chrono::time_point_cast (const time_point< _Clock, _Dur > &__t)`

6.73.1 Detailed Description

This is a Standard C++ Library header.

6.73.2 Typedef Documentation

6.73.2.1 `using std::chrono::_V2::high_resolution_clock = typedef system_clock`

Highest-resolution clock.

This is the clock "with the shortest tick period." Alias to `std::system_clock` until higher-than-nanosecond definitions become feasible.

Definition at line 776 of file `chrono`.

6.74 chrono File Reference

Namespaces

- [std](#)
- [std::chrono](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_CHRONO`

Variables

- `template<typename _Rep >`
`constexpr bool std::chrono::experimental::fundamentals_v1::treat_as_floating_point_v`

6.74.1 Detailed Description

This is a TS C++ Library header.

6.75 cinttypes File Reference

Macros

- `#define _GLIBCXX_CINTTYPES`

6.75.1 Detailed Description

This is a Standard C++ Library header.

6.76 cinttypes File Reference

Macros

- `#define _GLIBCXX_TR1_CINTTYPES`

6.76.1 Detailed Description

This is a TR1 C++ Library header.

6.77 ciso646 File Reference

6.77.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `iso646.h`, which is empty in C++.

6.78 climits File Reference

Macros

- `#define _GLIBCXX_CLIMITS`
- `#define LLONG_MAX`
- `#define LLONG_MIN`
- `#define ULLONG_MAX`

6.78.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the *.h implementation files.

This is the C++ version of the Standard C Library header `limits.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

6.79 climits File Reference

Macros

- `#define _GLIBCXX_TR1_CLIMITS`

6.79.1 Detailed Description

This is a TR1 C++ Library header.

6.80 clocale File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_CLOCALE`

6.80.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the *.h implementation files.

This is the C++ version of the Standard C Library header `locale.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

6.81 cmath File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_CMATH`
- `#define _GLIBCXX_INCLUDE_NEXT_C_HEADERS`

Functions

- constexpr double **std::abs** (double __x)
- constexpr float **std::abs** (float __x)
- constexpr long double **std::abs** (long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::abs** (_Tp __x)
- constexpr float **std::acos** (float __x)
- constexpr long double **std::acos** (long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::acos** (_Tp __x)
- constexpr float **std::asin** (float __x)
- constexpr long double **std::asin** (long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::asin** (_Tp __x)
- constexpr float **std::atan** (float __x)
- constexpr long double **std::atan** (long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::atan** (_Tp __x)
- constexpr float **std::atan2** (float __y, float __x)
- constexpr long double **std::atan2** (long double __y, long double __x)
- template<typename _Tp, typename _Up >
constexpr __gnu_cxx::__promote_2< _Tp, _Up >::__type **std::atan2** (_Tp __y, _Up __x)
- constexpr float **std::ceil** (float __x)
- constexpr long double **std::ceil** (long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::ceil** (_Tp __x)
- constexpr float **std::cos** (float __x)
- constexpr long double **std::cos** (long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::cos** (_Tp __x)
- constexpr float **std::cosh** (float __x)
- constexpr long double **std::cosh** (long double __x)
- template<typename _Tp >
constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type **std::cosh** (_Tp __x)
- constexpr float **std::exp** (float __x)
- constexpr long double **std::exp** (long double __x)

- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type std::exp (_Tp __x)`
- `constexpr float std::fabs (float __x)`
- `constexpr long double std::fabs (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type std::fabs (_Tp __x)`
- `constexpr float std::floor (float __x)`
- `constexpr long double std::floor (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type std::floor (_Tp __x)`
- `constexpr float std::fmod (float __x, float __y)`
- `constexpr long double std::fmod (long double __x, long double __y)`
- `template<typename _Tp, typename _Up >`
`constexpr __gnu_cxx::__promote_2< _Tp, _Up >::__type std::fmod (_Tp __x, _Up __y)`
- `float std::frexp (float __x, int * __exp)`
- `long double std::frexp (long double __x, int * __exp)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type std::frexp (_Tp __x, int * __exp)`
- `constexpr float std::ldexp (float __x, int __exp)`
- `constexpr long double std::ldexp (long double __x, int __exp)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type std::ldexp (_Tp __x, int __exp)`
- `constexpr float std::log (float __x)`
- `constexpr long double std::log (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type std::log (_Tp __x)`
- `constexpr float std::log10 (float __x)`
- `constexpr long double std::log10 (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type std::log10 (_Tp __x)`
- `float std::modf (float __x, float * __iptr)`
- `long double std::modf (long double __x, long double * __iptr)`
- `constexpr float std::pow (float __x, float __y)`
- `constexpr long double std::pow (long double __x, long double __y)`
- `template<typename _Tp, typename _Up >`
`constexpr __gnu_cxx::__promote_2< _Tp, _Up >::__type std::pow (_Tp __x, _Up __y)`
- `constexpr float std::sin (float __x)`
- `constexpr long double std::sin (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type std::sin (_Tp __x)`
- `constexpr float std::sinh (float __x)`
- `constexpr long double std::sinh (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type std::sinh (_Tp __x)`
- `constexpr float std::sqrt (float __x)`
- `constexpr long double std::sqrt (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type std::sqrt (_Tp __x)`
- `constexpr float std::tan (float __x)`
- `constexpr long double std::tan (long double __x)`

- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type std::tan (_Tp __x)`
- `constexpr float std::tanh (float __x)`
- `constexpr long double std::tanh (long double __x)`
- `template<typename _Tp >`
`constexpr __gnu_cxx::__enable_if< __is_integer< _Tp >::__value, double >::__type std::tanh (_Tp __x)`

6.81.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the *.h implementation files.

This is the C++ version of the Standard C Library header `math.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

6.82 cmath File Reference

Namespaces

- [__gnu_cxx](#)

Macros

- `#define _EXT_CMATH`

6.82.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.83 cmath File Reference

Namespaces

- [std](#)
- [std::tr1](#)

Macros

- `#define _GLIBCXX_TR1_CMATH`

Functions

- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::tr1::assoc_laguerre` (unsigned int __n, unsigned int __m, _Tp __x)
- `float std::tr1::assoc_laguerref` (unsigned int __n, unsigned int __m, float __x)
- `long double std::tr1::assoc_laguerrel` (unsigned int __n, unsigned int __m, long double __x)
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::tr1::assoc_legendre` (unsigned int __l, unsigned int __m, _Tp __x)
- `float std::tr1::assoc_legendref` (unsigned int __l, unsigned int __m, float __x)
- `long double std::tr1::assoc_legendrel` (unsigned int __l, unsigned int __m, long double __x)
- `template<typename _Tpx, typename _Tpy >`
`__gnu_cxx::__promote_2< _Tpx, _Tpy >::__type std::tr1::beta` (_Tpx __x, _Tpy __y)
- `float std::tr1::betaf` (float __x, float __y)
- `long double std::tr1::betal` (long double __x, long double __y)
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::tr1::comp_ellint_1` (_Tp __k)
- `float std::tr1::comp_ellint_1f` (float __k)
- `long double std::tr1::comp_ellint_1l` (long double __k)
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::tr1::comp_ellint_2` (_Tp __k)
- `float std::tr1::comp_ellint_2f` (float __k)
- `long double std::tr1::comp_ellint_2l` (long double __k)
- `template<typename _Tp, typename _Tpn >`
`__gnu_cxx::__promote_2< _Tp, _Tpn >::__type std::tr1::comp_ellint_3` (_Tp __k, _Tpn __nu)
- `float std::tr1::comp_ellint_3f` (float __k, float __nu)
- `long double std::tr1::comp_ellint_3l` (long double __k, long double __nu)
- `template<typename _Tpa, typename _Tpc, typename _Tp >`
`__gnu_cxx::__promote_3< _Tpa, _Tpc, _Tp >::__type std::tr1::conf_hyperg` (_Tpa __a, _Tpc __c, _Tp __x)
- `float std::tr1::conf_hypergf` (float __a, float __c, float __x)
- `long double std::tr1::conf_hypergl` (long double __a, long double __c, long double __x)
- `template<typename _Tpnu, typename _Tp >`
`__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::tr1::cyl_bessel_i` (_Tpnu __nu, _Tp __x)
- `float std::tr1::cyl_bessel_if` (float __nu, float __x)
- `long double std::tr1::cyl_bessel_il` (long double __nu, long double __x)
- `template<typename _Tpnu, typename _Tp >`
`__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::tr1::cyl_bessel_j` (_Tpnu __nu, _Tp __x)
- `float std::tr1::cyl_bessel_jf` (float __nu, float __x)
- `long double std::tr1::cyl_bessel_jl` (long double __nu, long double __x)
- `template<typename _Tpnu, typename _Tp >`
`__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::tr1::cyl_bessel_k` (_Tpnu __nu, _Tp __x)
- `float std::tr1::cyl_bessel_kf` (float __nu, float __x)
- `long double std::tr1::cyl_bessel_kl` (long double __nu, long double __x)
- `template<typename _Tpnu, typename _Tp >`
`__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::tr1::cyl_neumann` (_Tpnu __nu, _Tp __x)
- `float std::tr1::cyl_neumannf` (float __nu, float __x)
- `long double std::tr1::cyl_neumannl` (long double __nu, long double __x)
- `template<typename _Tp, typename _Tpp >`
`__gnu_cxx::__promote_2< _Tp, _Tpp >::__type std::tr1::ellint_1` (_Tp __k, _Tpp __phi)
- `float std::tr1::ellint_1f` (float __k, float __phi)
- `long double std::tr1::ellint_1l` (long double __k, long double __phi)
- `template<typename _Tp, typename _Tpp >`
`__gnu_cxx::__promote_2< _Tp, _Tpp >::__type std::tr1::ellint_2` (_Tp __k, _Tpp __phi)

- float **std::tr1::ellint_2f** (float __k, float __phi)
- long double **std::tr1::ellint_2l** (long double __k, long double __phi)
- template<typename _Tp, typename _Tpn, typename _Tpp >
__gnu_cxx::__promote_3< _Tp, _Tpn, _Tpp >::__type **std::tr1::ellint_3** (_Tp __k, _Tpn __nu, _Tpp __phi)
- float **std::tr1::ellint_3f** (float __k, float __nu, float __phi)
- long double **std::tr1::ellint_3l** (long double __k, long double __nu, long double __phi)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::expint** (_Tp __x)
- float **std::tr1::expintf** (float __x)
- long double **std::tr1::expintl** (long double __x)
- float **std::tr1::fabs** (float __x)
- long double **std::tr1::fabs** (long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::fabs** (_Tp __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::hermite** (unsigned int __n, _Tp __x)
- float **std::tr1::hermitef** (unsigned int __n, float __x)
- long double **std::tr1::hermitel** (unsigned int __n, long double __x)
- template<typename _Tpa, typename _Tpb, typename _Tpc, typename _Tp >
__gnu_cxx::__promote_4< _Tpa, _Tpb, _Tpc, _Tp >::__type **std::tr1::hyperg** (_Tpa __a, _Tpb __b, _Tpc __c, _Tp __x)
- float **std::tr1::hypergf** (float __a, float __b, float __c, float __x)
- long double **std::tr1::hypergl** (long double __a, long double __b, long double __c, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::laguerre** (unsigned int __n, _Tp __x)
- float **std::tr1::laguerref** (unsigned int __n, float __x)
- long double **std::tr1::laguerrel** (unsigned int __n, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::legendre** (unsigned int __n, _Tp __x)
- float **std::tr1::legendref** (unsigned int __n, float __x)
- long double **std::tr1::legendrel** (unsigned int __n, long double __x)
- float **std::tr1::pow** (float __x, float __y)
- long double **std::tr1::pow** (long double __x, long double __y)
- template<typename _Tp, typename _Up >
__gnu_cxx::__promote_2< _Tp, _Up >::__type **std::tr1::pow** (_Tp __x, _Up __y)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::riemann_zeta** (_Tp __x)
- float **std::tr1::riemann_zetaf** (float __x)
- long double **std::tr1::riemann_zetal** (long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::sph_bessel** (unsigned int __n, _Tp __x)
- float **std::tr1::sph_besself** (unsigned int __n, float __x)
- long double **std::tr1::sph_bessell** (unsigned int __n, long double __x)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::sph_legendre** (unsigned int __l, unsigned int __m, _Tp __theta)
- float **std::tr1::sph_legendref** (unsigned int __l, unsigned int __m, float __theta)
- long double **std::tr1::sph_legendrel** (unsigned int __l, unsigned int __m, long double __theta)
- template<typename _Tp >
__gnu_cxx::__promote< _Tp >::__type **std::tr1::sph_neumann** (unsigned int __n, _Tp __x)
- float **std::tr1::sph_neumannf** (unsigned int __n, float __x)
- long double **std::tr1::sph_neumannl** (unsigned int __n, long double __x)

6.83.1 Detailed Description

This is a TR1 C++ Library header.

6.84 `cmp_fn_imps.hpp` File Reference

6.84.1 Detailed Description

Contains implementations of `cc_ht_map_'s` entire container comparison related functions.

6.85 `codecvt` File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_CODECVT`
- `#define _GLIBCXX_CODECVT_SPECIALIZATION(_NAME, _ELEM)`
- `#define _GLIBCXX_CODECVT_SPECIALIZATION2(_NAME, _ELEM)`

Enumerations

- enum `codecvt_mode` { `consume_header`, `generate_header`, `little_endian` }

6.85.1 Detailed Description

This is a Standard C++ Library header.

6.86 `codecvt.h` File Reference

Classes

- class [std::__codecvt_abstract_base< _InternT, _ExternT, _StateT >](#)
- class [std::codecvt< _InternT, _ExternT, _StateT >](#)
- class [std::codecvt< char, char, mbstate_t >](#)
- class [std::codecvt< char16_t, char, mbstate_t >](#)
- class [std::codecvt< char32_t, char, mbstate_t >](#)
- class [std::codecvt< wchar_t, char, mbstate_t >](#)
- class [std::codecvt_base](#)
- class [std::codecvt_byname< _InternT, _ExternT, _StateT >](#)

Namespaces

- [std](#)

6.86.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

6.87 `codecvt_specializations.h` File Reference

Classes

- struct [__gnu_cxx::encoding_char_traits<_CharT>](#)
- class [__gnu_cxx::encoding_state](#)
- class [std::codecvt<_InternT, _ExternT, encoding_state>](#)

Namespaces

- [__gnu_cxx](#)
- [std](#)

Functions

- template<typename _Tp>
size_t **std::__iconv_adapter** (size_t(*__func)(iconv_t, _Tp, size_t *, char **, size_t *), iconv_t __cd, char **__
__inbuf, size_t *__inbytes, char **__outbuf, size_t *__outbytes)

6.87.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.88 `compatibility.h` File Reference

6.88.1 Detailed Description

This is an internal header file, included by other library sources. You should not attempt to use it directly.

6.89 `compatibility.h` File Reference

Namespaces

- [__gnu_parallel](#)

Functions

- `template<typename _Tp >`
`_Tp __gnu_parallel::__add_omp (volatile _Tp * __ptr, _Tp __addend)`
- `template<typename _Tp >`
`bool __gnu_parallel::__cas_omp (volatile _Tp * __ptr, _Tp __comparand, _Tp __replacement)`
- `template<typename _Tp >`
`bool __gnu_parallel::__compare_and_swap (volatile _Tp * __ptr, _Tp __comparand, _Tp __replacement)`
- `template<typename _Tp >`
`_Tp __gnu_parallel::__fetch_and_add (volatile _Tp * __ptr, _Tp __addend)`
- `void __gnu_parallel::__yield ()`

6.89.1 Detailed Description

Compatibility layer, mostly concerned with atomic operations.

This file is a GNU parallel extension to the Standard C++ Library and contains implementation details for the library's internal use.

6.90 compiletime_settings.h File Reference

Macros

- `#define _GLIBCXX_CALL(__n)`
- `#define _GLIBCXX_RANDOM_SHUFFLE_CONSIDER_L1`
- `#define _GLIBCXX_RANDOM_SHUFFLE_CONSIDER_TLB`
- `#define _GLIBCXX_SCALE_DOWN_FPU`
- `#define _GLIBCXX_VERBOSE_LEVEL`

6.90.1 Detailed Description

Defines on options concerning debugging and performance, at compile-time. This file is a GNU parallel extension to the Standard C++ Library.

6.90.2 Macro Definition Documentation

6.90.2.1 `#define _GLIBCXX_CALL(__n)`

Macro to produce log message when entering a function.

Parameters

<code>__↔</code>	Input size.
<code>__n</code>	

See also

`_GLIBCXX_VERBOSE_LEVEL`

Definition at line 44 of file `comPILEtime_settings.h`.

Referenced by `__gnu_parallel::__find_template()`, `__gnu_parallel::__for_each_template_random_access_workstealing()`, `__gnu_parallel::__merge_advance()`, `__gnu_parallel::__parallel_nth_element()`, `__gnu_parallel::__parallel_partial_sum()`, `__gnu_parallel::__parallel_partition()`, `__gnu_parallel::__parallel_random_shuffle_drs()`, `__gnu_parallel::__parallel_sort()`, `__gnu_parallel::__parallel_sort_qs()`, `__gnu_parallel::__parallel_sort_qsb()`, `__gnu_parallel::__parallel_unique_copy()`, `__gnu_parallel::__search_template()`, `__gnu_parallel::__sequential_multiway_merge()`, `__gnu_parallel::__multiseq_partition()`, `__gnu_parallel::__multiseq_selection()`, `__gnu_parallel::__multiway_merge()`, `__gnu_parallel::__multiway_merge_3_variant()`, `__gnu_parallel::__multiway_merge_4_variant()`, `__gnu_parallel::__multiway_merge_loser_tree()`, `__gnu_parallel::__multiway_merge_loser_tree_sentinel()`, `__gnu_parallel::__multiway_merge_loser_tree_unguarded()`, `__gnu_parallel::__multiway_merge_sentinels()`, `__gnu_parallel::__parallel_multiway_merge()`, and `__gnu_parallel::__parallel_sort_mwms()`.

6.90.2.2 `#define _GLIBCXX_RANDOM_SHUFFLE_CONSIDER_L1`

Switch on many `_GLIBCXX_PARALLEL_ASSERTions` in parallel code. Consider the size of the L1 cache for `__gnu_parallel::__parallel_random_shuffle()`.

Definition at line 62 of file `comPILEtime_settings.h`.

6.90.2.3 `#define _GLIBCXX_RANDOM_SHUFFLE_CONSIDER_TLB`

Switch on many `_GLIBCXX_PARALLEL_ASSERTions` in parallel code. Consider the size of the TLB for `__gnu_parallel::__parallel_random_shuffle()`.

Definition at line 68 of file `comPILEtime_settings.h`.

6.90.2.4 `#define _GLIBCXX_SCALE_DOWN_FPU`

Use floating-point scaling instead of modulo for mapping random numbers to a range. This can be faster on certain CPUs.

Definition at line 55 of file `comPILEtime_settings.h`.

6.90.2.5 `#define _GLIBCXX_VERBOSE_LEVEL`

Determine verbosity level of the parallel mode. Level 1 prints a message each time a parallel-mode function is entered.

Definition at line 37 of file `comPILEtime_settings.h`.

6.91 complex File Reference

Classes

- struct `std::complex< _Tp >`
- struct `std::complex< _Tp >`
- struct `std::complex< double >`
- struct `std::complex< float >`
- struct `std::complex< long double >`

Namespaces

- [__gnu_cxx](#)
- [std](#)

Macros

- `#define __cpp_lib_complex_udls`
- `#define _GLIBCXX_COMPLEX`

Functions

- `template<typename _Tp >`
`_Tp std::__complex_abs (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > std::__complex_acos (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > std::__complex_acosh (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`_Tp std::__complex_arg (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > std::__complex_asin (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > std::__complex_asinh (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > std::__complex_atan (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > std::__complex_atanh (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > std::__complex_cos (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > std::__complex_cosh (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > std::__complex_exp (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > std::__complex_log (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > std::__complex_pow (const complex< _Tp > &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`complex< _Tp > std::__complex_pow_unsigned (complex< _Tp > __x, unsigned __n)`
- `template<typename _Tp >`
`std::complex< _Tp > std::__complex_proj (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > std::__complex_sin (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > std::__complex_sinh (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > std::__complex_sqrt (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > std::__complex_tan (const complex< _Tp > &__z)`

- `template<typename _Tp >`
`complex< _Tp > std::__complex_tanh (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`_Tp std::abs (const complex< _Tp > &)`
- `template<typename _Tp >`
`std::complex< _Tp > std::acos (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > std::acosh (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`_Tp std::arg (const complex< _Tp > &)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::arg (_Tp __x)`
- `template<typename _Tp >`
`std::complex< _Tp > std::asin (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > std::asinh (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > std::atan (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< _Tp > std::atanh (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`complex< _Tp > std::conj (const complex< _Tp > &)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::conj (_Tp __x)`
- `template<typename _Tp >`
`complex< _Tp > std::cos (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::cosh (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::exp (const complex< _Tp > &)`
- `template<typename _Tp >`
`_Tp std::fabs (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`constexpr _Tp std::imag (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::imag (_Tp)`
- `template<typename _Tp >`
`complex< _Tp > std::log (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::log10 (const complex< _Tp > &)`
- `template<typename _Tp >`
`_Tp std::norm (const complex< _Tp > &)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::norm (_Tp __x)`
- `constexpr std::complex< double > std::literals::complex_literals::operator""i (long double __num)`
- `constexpr std::complex< double > std::literals::complex_literals::operator""i (unsigned long long __num)`
- `constexpr std::complex< float > std::literals::complex_literals::operator""if (long double __num)`
- `constexpr std::complex< float > std::literals::complex_literals::operator""if (unsigned long long __num)`
- `constexpr std::complex< long double > std::literals::complex_literals::operator""il (long double __num)`
- `constexpr std::complex< long double > std::literals::complex_literals::operator""il (unsigned long long __num)`

- `template<typename _Tp >`
`complex< _Tp > std::operator+ (const complex< _Tp > &__x)`
- `template<typename _Tp >`
`complex< _Tp > std::operator- (const complex< _Tp > &__x)`
- `template<typename _Tp, typename _CharT, class _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, const complex< _Tp > &__x)`
- `template<typename _Tp, typename _CharT, class _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__is, complex< _Tp > &__x)`
- `template<typename _Tp >`
`complex< _Tp > std::polar (const _Tp &, const _Tp &=0)`
- `template<typename _Tp >`
`complex< _Tp > std::pow (const complex< _Tp > &, int)`
- `template<typename _Tp >`
`complex< _Tp > std::pow (const complex< _Tp > &, const _Tp &)`
- `template<typename _Tp >`
`complex< _Tp > std::pow (const complex< _Tp > &, const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::pow (const _Tp &, const complex< _Tp > &)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::pow (const std::complex< _Tp > &__x, const _Up &__y)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::pow (const _Tp &__x, const std::complex< _Up > &__y)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::pow (const std::complex< _Tp > &__x, const std::complex< _Up > &__y)`
- `template<typename _Tp >`
`std::complex< _Tp > std::proj (const std::complex< _Tp > &)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::proj (_Tp __x)`
- `template<typename _Tp >`
`constexpr _Tp std::real (const complex< _Tp > &__z)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::real (_Tp __x)`
- `template<typename _Tp >`
`complex< _Tp > std::sin (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::sinh (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::sqrt (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::tan (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::tanh (const complex< _Tp > &)`
- `template<typename _Tp >`
`complex< _Tp > std::operator+ (const complex< _Tp > &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`complex< _Tp > std::operator+ (const complex< _Tp > &__x, const _Tp &__y)`

- `template<typename _Tp >`
`complex< _Tp > std::operator+ (const _Tp &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`complex< _Tp > std::operator- (const complex< _Tp > &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`complex< _Tp > std::operator- (const complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp >`
`complex< _Tp > std::operator- (const _Tp &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`complex< _Tp > std::operator* (const complex< _Tp > &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`complex< _Tp > std::operator* (const complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp >`
`complex< _Tp > std::operator* (const _Tp &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`complex< _Tp > std::operator/ (const complex< _Tp > &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`complex< _Tp > std::operator/ (const complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp >`
`complex< _Tp > std::operator/ (const _Tp &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`constexpr bool std::operator== (const complex< _Tp > &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`constexpr bool std::operator== (const complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp >`
`constexpr bool std::operator== (const _Tp &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`constexpr bool std::operator!= (const complex< _Tp > &__x, const complex< _Tp > &__y)`
- `template<typename _Tp >`
`constexpr bool std::operator!= (const complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp >`
`constexpr bool std::operator!= (const _Tp &__x, const complex< _Tp > &__y)`

6.91.1 Detailed Description

This is a Standard C++ Library header.

6.92 complex File Reference

Namespaces

- [std](#)
- [std::tr1](#)

Macros

- `#define _GLIBCXX_TR1_COMPLEX`

Functions

- `template<typename _Tp >`
`std::complex< _Tp > std::tr1::conj (const std::complex< _Tp > &__z)`
- `template<typename _Tp >`
`std::complex< typename __gnu_cxx::__promote< _Tp >::__type > std::tr1::conj (_Tp __x)`
- `template<typename _Tp >`
`std::complex< _Tp > std::tr1::fabs (const std::complex< _Tp > &__z)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::tr1::polar (const _Tp &__rho,`
`const _Up &__theta)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::tr1::pow (const std::complex<`
`_Tp > &__x, const _Up &__y)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::tr1::pow (const _Tp &__x,`
`const std::complex< _Up > &__y)`
- `template<typename _Tp, typename _Up >`
`std::complex< typename __gnu_cxx::__promote_2< _Tp, _Up >::__type > std::tr1::pow (const std::complex<`
`_Tp > &__x, const std::complex< _Up > &__y)`
- `template<typename _Tp >`
`std::complex< _Tp > std::tr1::pow (const std::complex< _Tp > &__x, const _Tp &__y)`
- `template<typename _Tp >`
`std::complex< _Tp > std::tr1::pow (const _Tp &__x, const std::complex< _Tp > &__y)`
- `template<typename _Tp >`
`std::complex< _Tp > std::tr1::pow (const std::complex< _Tp > &__x, const std::complex< _Tp > &__y)`

6.92.1 Detailed Description

This is a TR1 C++ Library header.

6.93 complex.h File Reference

Macros

- `#define _GLIBCXX_COMPLEX_H`

6.93.1 Detailed Description

This is a Standard C++ Library header.

6.94 `concept_check.h` File Reference

Macros

- `#define __glibcxx_class_requires(_a, _b)`
- `#define __glibcxx_class_requires2(_a, _b, _c)`
- `#define __glibcxx_class_requires3(_a, _b, _c, _d)`
- `#define __glibcxx_class_requires4(_a, _b, _c, _d, _e)`
- `#define __glibcxx_function_requires(...)`

6.94.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iterator>`.

6.95 `concurrency.h` File Reference

Classes

- class [__gnu_cxx::__scoped_lock](#)

Namespaces

- [__gnu_cxx](#)

Enumerations

- enum `_Lock_policy` { `_S_single`, `_S_mutex`, `_S_atomic` }

Functions

- void `__gnu_cxx::__throw_concurrency_lock_error()`
- void `__gnu_cxx::__throw_concurrency_unlock_error()`

Variables

- static const `_Lock_policy` `__gnu_cxx::__default_lock_policy`

6.95.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.96 cond_dealtor.hpp File Reference

Classes

- class [__gnu_pbds::detail::cond_dealtor< Entry, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

6.96.1 Detailed Description

Contains a conditional deallocator.

6.97 cond_key_dtor_entry_dealtor.hpp File Reference

Classes

- class [__gnu_pbds::detail::cond_dealtor< Entry, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

6.97.1 Detailed Description

Contains a conditional key destructor, used for exception handling.

6.98 condition_variable File Reference

Classes

- class [std::_V2::condition_variable_any](#)
- class [std::condition_variable](#)

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_CONDITION_VARIABLE`

Enumerations

- enum [std::cv_status](#) { **no_timeout**, **timeout** }

Functions

- void **std::notify_all_at_thread_exit** (condition_variable &, unique_lock< mutex >)

6.98.1 Detailed Description

This is a Standard C++ Library header.

6.99 [const_iterator.hpp](#) File Reference

Classes

- class [__gnu_pbds::detail::binary_heap_const_iterator_](#)< Value_Type, Entry, Simple, _Alloc >

Namespaces

- [__gnu_pbds](#)

Macros

- #define **PB_DS_BIN_HEAP_CIT_BASE**

6.99.1 Detailed Description

Contains an iterator class returned by the table's const find and insert methods.

6.100 [const_iterator.hpp](#) File Reference

Classes

- class [__gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_](#)< Node, _Alloc >

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_BASIC_HEAP_CIT_BASE`
- `#define PB_DS_CLASS_C_DEC`

6.100.1 Detailed Description

Contains an iterator class returned by the table's `const find` and `insert` methods.

6.101 `const_iterator.hpp` File Reference

Classes

- class [const_iterator_](#)

6.101.1 Detailed Description

Contains an iterator class used for `const` ranging over the elements of the table.

6.102 `constructor_destructor_fn_imps.hpp` File Reference

6.102.1 Detailed Description

Contains implementations of `cc_ht_map_`'s constructors, destructor, and related functions.

6.103 `constructor_destructor_fn_imps.hpp` File Reference

6.103.1 Detailed Description

Contains implementations of `gp_ht_map_`'s constructors, destructor, and related functions.

6.104 `constructor_destructor_fn_imps.hpp` File Reference6.105 `constructor_destructor_no_store_hash_fn_imps.hpp` File Reference

6.105.1 Detailed Description

Contains implementations of `cc_ht_map_`'s constructors, destructor, and related functions.

6.106 constructor_destructor_no_store_hash_fn_imps.hpp File Reference

6.106.1 Detailed Description

Contains implementations of gp_ht_map_'s constructors, destructor, and related functions.

6.107 constructor_destructor_store_hash_fn_imps.hpp File Reference

6.107.1 Detailed Description

Contains implementations of cc_ht_map_'s constructors, destructor, and related functions.

6.108 constructor_destructor_store_hash_fn_imps.hpp File Reference

6.108.1 Detailed Description

Contains implementations of gp_ht_map_'s constructors, destructor, and related functions.

6.109 constructors_destructor_fn_imps.hpp File Reference

6.109.1 Detailed Description

Contains an implementation class for binary_heap_.

6.110 constructors_destructor_fn_imps.hpp File Reference

6.110.1 Detailed Description

Contains an implementation for binomial_heap_.

6.111 constructors_destructor_fn_imps.hpp File Reference

6.111.1 Detailed Description

Contains an implementation class for a base of binomial heaps.

6.112 constructors_destructor_fn_imps.hpp File Reference

6.112.1 Detailed Description

Contains an implementation class for bin_search_tree_.

6.113 constructors_destructor_fn_imps.hpp File Reference

6.113.1 Detailed Description

Contains an implementation class for left_child_next_sibling_heap_.

6.114 constructors_destructor_fn_imps.hpp File Reference

6.114.1 Detailed Description

Contains an implementation class for ov_tree_.

6.115 constructors_destructor_fn_imps.hpp File Reference

6.115.1 Detailed Description

Contains an implementation class for a pairing heap.

6.116 constructors_destructor_fn_imps.hpp File Reference

6.116.1 Detailed Description

Contains an implementation class for pat_trie.

6.117 constructors_destructor_fn_imps.hpp File Reference

6.117.1 Detailed Description

Contains an implementation for rb_tree_.

6.118 constructors_destructor_fn_imps.hpp File Reference

6.118.1 Detailed Description

Contains an implementation for rc_binomial_heap_.

6.119 constructors_destructor_fn_imps.hpp File Reference

6.119.1 Detailed Description

Contains an implementation class for splay_tree_.

6.120 constructors_destructor_fn_imps.hpp File Reference

6.120.1 Detailed Description

Contains an implementation for thin_heap_.

6.121 container_base_dispatch.hpp File Reference

Classes

- [struct `__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, cc_hash_tag, Policy_TI >`](#)
- [struct `__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, gp_hash_tag, Policy_TI >`](#)
- [struct `__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, list_update_tag, Policy_TI >`](#)
- [struct `__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, ov_tree_tag, Policy_TI >`](#)
- [struct `__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, pat_trie_tag, Policy_TI >`](#)
- [struct `__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, rb_tree_tag, Policy_TI >`](#)
- [struct `__gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, splay_tree_tag, Policy_TI >`](#)
- [struct `__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, cc_hash_tag, Policy_TI >`](#)
- [struct `__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, gp_hash_tag, Policy_TI >`](#)
- [struct `__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, list_update_tag, Policy_TI >`](#)
- [struct `__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, ov_tree_tag, Policy_TI >`](#)
- [struct `__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, pat_trie_tag, Policy_TI >`](#)
- [struct `__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, rb_tree_tag, Policy_TI >`](#)
- [struct `__gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, splay_tree_tag, Policy_TI >`](#)

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_ASSERT_VALID(X)`
- `#define PB_DS_CHECK_KEY_DOES_NOT_EXIST(_Key)`
- `#define PB_DS_CHECK_KEY_EXISTS(_Key)`
- `#define PB_DS_DATA_FALSE_INDICATOR`
- `#define PB_DS_DATA_TRUE_INDICATOR`
- `#define PB_DS_DEBUG_VERIFY(_Cond)`
- `#define PB_DS_EP2VP(X)`
- `#define PB_DS_EP2VP(X)`
- `#define PB_DS_V2F(X)`
- `#define PB_DS_V2F(X)`
- `#define PB_DS_V2S(X)`
- `#define PB_DS_V2S(X)`

6.121.1 Detailed Description

Contains associative container dispatching.

6.122 `cpp_type_traits.h` File Reference

Namespaces

- [std](#)

Macros

- `#define __INT_N(TYPE)`

Functions

- `template<typename _Iterator >
_Iterator std::__miter_base (_Iterator __it)`

6.122.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ext/type_traits>`.

6.123 `cpu_defines.h` File Reference

6.123.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iosfwd>`.

6.124 csetjmp File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_CSETJMP`
- `#define setjmp(env)`

6.124.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `setjmp.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

6.125 csignal File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_CSIGNAL`

6.125.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `signal.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

6.126 cstdalign File Reference

Macros

- `#define _GLIBCXX_CSTDALIGN`

6.126.1 Detailed Description

This is a Standard C++ Library header.

6.127 cstdarg File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_CSTDARG`
- `#define va_end(ap)`

6.127.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `stdarg.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

6.128 cstdarg File Reference

Macros

- `#define _GLIBCXX_TR1_CSTDARG`

6.128.1 Detailed Description

This is a TR1 C++ Library header.

6.129 cstdlib File Reference

Macros

- `#define _GLIBCXX_CSTDBOOL`

6.129.1 Detailed Description

This is a Standard C++ Library header.

6.130 cstdlib File Reference

Macros

- `#define _GLIBCXX_TR1_CSTDBOOL`

6.130.1 Detailed Description

This is a TR1 C++ Library header.

6.131 cstdint File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_CSTDDEF`

6.131.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `stdint.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

6.132 cstdint File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_CSTDINT`

6.132.1 Detailed Description

This is a Standard C++ Library header.

6.133 cstdint File Reference

Namespaces

- [std](#)
- [std::tr1](#)

Macros

- `#define _GLIBCXX_TR1_CSTDINT`

6.133.1 Detailed Description

This is a TR1 C++ Library header.

6.134 cstdio File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_CSTDIO`

6.134.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `stdio.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

6.135 cstdio File Reference

Macros

- `#define _GLIBCXX_TR1_CSTDIO`

6.135.1 Detailed Description

This is a TR1 C++ Library header.

6.136 cstdlib File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_CSTDlib`
- `#define EXIT_FAILURE`
- `#define EXIT_SUCCESS`

Functions

- void **std::abort** (void) throw ()
- int **std::atexit** (void*)(void)) throw ()
- void **std::exit** (int) throw ()

6.136.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `stdlib.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

6.137 cstdlib File Reference

Macros

- `#define _GLIBCXX_TR1_CSTDlib`

6.137.1 Detailed Description

This is a TR1 C++ Library header.

6.138 cstring File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_CSTRING`

Functions

- `void * std::memchr (void *__s, int __c, size_t __n)`
- `char * std::strchr (char *__s, int __n)`
- `char * std::strpbrk (char *__s1, const char *__s2)`
- `char * std::strrchr (char *__s, int __n)`
- `char * std::strstr (char *__s1, const char *__s2)`

6.138.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `string.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

6.139 `ctgmath` File Reference

Macros

- `#define _GLIBCXX_CTGMATH`

6.139.1 Detailed Description

This is a Standard C++ Library header.

6.140 `ctgmath` File Reference

Macros

- `#define _GLIBCXX_TR1_CTGMATH`

6.140.1 Detailed Description

This is a TR1 C++ Library header.

6.141 ctime File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_CTIME`

6.141.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the *.h implementation files.

This is the C++ version of the Standard C Library header `time.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

6.142 ctime File Reference

Macros

- `#define _GLIBCXX_TR1_CTIME`

6.142.1 Detailed Description

This is a TR1 C++ Library header.

6.143 ctype_base.h File Reference

Classes

- struct [std::ctype_base](#)

Namespaces

- [std](#)

6.143.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

6.144 ctype_inline.h File Reference

Namespaces

- [std](#)

6.144.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

6.145 cuchar File Reference

Macros

- `#define _GLIBCXX_CUCHAR`

6.145.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `uchar.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

6.146 cwchar File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_CWCHAR`

Functions

- `wchar_t * std::wcschr (wchar_t *__p, wchar_t __c)`
- `wchar_t * std::wcsprk (wchar_t *__s1, const wchar_t *__s2)`
- `wchar_t * std::wcsrchr (wchar_t *__p, wchar_t __c)`
- `wchar_t * std::wcsstr (wchar_t *__s1, const wchar_t *__s2)`
- `wchar_t * std::wmemchr (wchar_t *__p, wchar_t __c, size_t __n)`

6.146.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `wchar.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

6.147 `cwchar` File Reference

Namespaces

- [std](#)
- [std::tr1](#)

Macros

- `#define _GLIBCXX_TR1_CWCHAR`

6.147.1 Detailed Description

This is a TR1 C++ Library header.

6.148 `cwctype` File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_CWCTYPE`

6.148.1 Detailed Description

This is a Standard C++ Library file. You should `#include` this file in your programs, rather than any of the `*.h` implementation files.

This is the C++ version of the Standard C Library header `wctype.h`, and its contents are (mostly) the same as that header, but are all contained in the namespace `std` (except for names which are defined as macros in C).

6.149 cwctype File Reference

Namespaces

- [std](#)
- [std::tr1](#)

Macros

- `#define _GLIBCXX_TR1_CWCTYPE`

6.149.1 Detailed Description

This is a TR1 C++ Library header.

6.150 cxxabi.h File Reference

Classes

- class [__gnu_cxx::recursive_init_error](#)

Namespaces

- [__gnu_cxx](#)
- [abi](#)

Typedefs

- typedef `__cxa_cdtor_return_type(* __cxxabiv1::__cxa_cdtor_type) (void *)`

Functions

- `__cxa_dependent_exception * __cxxabiv1::__cxa_allocate_dependent_exception () noexcept`
- `void * __cxxabiv1::__cxa_allocate_exception (size_t) noexcept`
- `int __cxxabiv1::__cxa_atexit (void (*)(void *), void *, void *) noexcept`
- `void __cxxabiv1::__cxa_bad_cast () __attribute__((__noreturn__))`
- `void __cxxabiv1::__cxa_bad_typeid () __attribute__((__noreturn__))`
- `void * __cxxabiv1::__cxa_begin_catch (void *) noexcept`
- `std::type_info * __cxxabiv1::__cxa_current_exception_type () noexcept __attribute__((__pure__))`
- `void __cxxabiv1::__cxa_deleted_virtual (void) __attribute__((__noreturn__))`
- `char * __cxxabiv1::__cxa_demangle (const char * __mangled_name, char * __output_buffer, size_t * __length, int * __status)`
- `void __cxxabiv1::__cxa_end_catch ()`
- `int __cxxabiv1::__cxa_finalize (void *)`

- void **__cxxabiv1::__cxa_free_dependent_exception** (__cxa_dependent_exception *) noexcept
- void **__cxxabiv1::__cxa_free_exception** (void *) noexcept
- void * **__cxxabiv1::__cxa_get_exception_ptr** (void *) noexcept __attribute__((__pure__))
- __cxa_eh_globals * **__cxxabiv1::__cxa_get_globals** () noexcept __attribute__((__const__))
- __cxa_eh_globals * **__cxxabiv1::__cxa_get_globals_fast** () noexcept __attribute__((__const__))
- void **__cxxabiv1::__cxa_guard_abort** (__guard *) noexcept
- int **__cxxabiv1::__cxa_guard_acquire** (__guard *)
- void **__cxxabiv1::__cxa_guard_release** (__guard *) noexcept
- void **__cxxabiv1::__cxa_pure_virtual** (void) __attribute__((__noreturn__))
- void **__cxxabiv1::__cxa_rethrow** () __attribute__((__noreturn__))
- int **__cxxabiv1::__cxa_thread_atexit** (void(*) (void *), void *, void *) noexcept
- void **__cxxabiv1::__cxa_throw** (void *, [std::type_info](#) *, void(*) (void *)) __attribute__((__noreturn__))
- void **__cxxabiv1::__cxa_throw_bad_array_new_length** () __attribute__((__noreturn__))
- __cxa_vec_ctor_return_type **__cxxabiv1::__cxa_vec_ctor** (void * __dest_array, void * __src_array, size_t __↵
element_count, size_t __element_size, __cxa_ctor_return_type(*) __constructor)(void *, void *), __cxa_ctor↵
__type __destructor)
- void **__cxxabiv1::__cxa_vec_cleanup** (void * __array_address, size_t __element_count, size_t __s, __cxa↵
__ctor_type __destructor) noexcept
- __cxa_vec_ctor_return_type **__cxxabiv1::__cxa_vec_ctor** (void * __array_address, size_t __element_count,
size_t __element_size, __cxa_ctor_type __constructor, __cxa_ctor_type __destructor)
- void **__cxxabiv1::__cxa_vec_delete** (void * __array_address, size_t __element_size, size_t __padding_size, ↵
__cxa_ctor_type __destructor)
- void **__cxxabiv1::__cxa_vec_delete2** (void * __array_address, size_t __element_size, size_t __padding_size,
__cxa_ctor_type __destructor, void(*) __dealloc)(void *)
- void **__cxxabiv1::__cxa_vec_delete3** (void * __array_address, size_t __element_size, size_t __padding_size,
__cxa_ctor_type __destructor, void(*) __dealloc)(void *, size_t)
- void **__cxxabiv1::__cxa_vec_dtor** (void * __array_address, size_t __element_count, size_t __element_size, ↵
__cxa_ctor_type __destructor)
- void * **__cxxabiv1::__cxa_vec_new** (size_t __element_count, size_t __element_size, size_t __padding_size,
__cxa_ctor_type __constructor, __cxa_ctor_type __destructor)
- void * **__cxxabiv1::__cxa_vec_new2** (size_t __element_count, size_t __element_size, size_t __padding_size,
__cxa_ctor_type __constructor, __cxa_ctor_type __destructor, void(*) __alloc)(size_t), void(*) __dealloc)(void
*)
- void * **__cxxabiv1::__cxa_vec_new3** (size_t __element_count, size_t __element_size, size_t __padding_size,
__cxa_ctor_type __constructor, __cxa_ctor_type __destructor, void(*) __alloc)(size_t), void(*) __dealloc)(void
*, size_t)
- void * **__cxxabiv1::__dynamic_cast** (const void * __src_ptr, const __class_type_info * __src_type, const __↵
class_type_info * __dst_type, ptrdiff_t __src2dst)

6.150.1 Detailed Description

The header provides an interface to the C++ ABI.

6.150.2 Function Documentation

- 6.150.2.1 **char* __cxxabiv1::__cxa_demangle** (const char * __mangled_name, char * __output_buffer, size_t * __length, int *
__status)

Demangling routine. ABI-mandated entry point in the C++ runtime library for demangling.

Parameters

<code>__mangled_name</code>	A NUL-terminated character string containing the name to be demangled.
<code>__output_buffer</code>	A region of memory, allocated with <code>malloc</code> , of <code>*__length</code> bytes, into which the demangled name is stored. If <code>__output_buffer</code> is not long enough, it is expanded using <code>realloc</code> . <code>__output_buffer</code> may instead be <code>NULL</code> ; in that case, the demangled name is placed in a region of memory allocated with <code>malloc</code> .
<code>__length</code>	If <code>__length</code> is non- <code>NULL</code> , the length of the buffer containing the demangled name is placed in <code>*__length</code> .
<code>__status</code>	<code>*__status</code> is set to one of the following values: 0: The demangling operation succeeded. -1: A memory allocation failure occurred. -2: <code>mangled_name</code> is not a valid name under the C++ ABI mangling rules. -3: One of the arguments is invalid.

Returns

A pointer to the start of the NUL-terminated demangled name, or `NULL` if the demangling fails. The caller is responsible for deallocating this memory using `free`.

The demangling is performed using the C++ ABI mangling rules, with GNU extensions. For example, this function is used in `__gnu_cxx::__verbose_terminate_handler`.

See <http://gcc.gnu.org/onlinedocs/libstdc++/manual/bk01pt12ch39.html> for other examples of use.

Note

The same demangling functionality is available via `libiberty` (`<libiberty/demangle.h>` and `libiberty.h`) in GCC 3.1 and later, but that requires explicit installation (`-enable-install-libiberty`) and uses a different API, although the ABI is unchanged.

6.151 cxxabi_forced.h File Reference

Classes

- class `__cxxabiv1::__forced_unwind`

6.151.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<cxxabi.h>`.

6.152 cxxabi_tweaks.h File Reference

Macros

- `#define _GLIBCXX_CXA_VEC_CTOR_RETURN(x)`
- `#define _GLIBCXX_GUARD_BIT`
- `#define _GLIBCXX_GUARD_PENDING_BIT`
- `#define _GLIBCXX_GUARD_SET(x)`
- `#define _GLIBCXX_GUARD_TEST(x)`
- `#define _GLIBCXX_GUARD_WAITING_BIT`

Typedefs

- typedef void `__cxxabiv1::__cxa_cdtor_return_type`
- typedef void `__cxxabiv1::__cxa_vec_ctor_return_type`

Functions

- `__extension__` typedef int `__guard` `__cxxabiv1::__attribute__` ((mode(__DI__)))

6.152.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<cxxabi.h>`.

6.153 debug.h File Reference

Namespaces

- [__gnu_debug](#)
- [std](#)
- [std::__debug](#)

Macros

- #define `__glibcxx_requires_cond`(_Cond, _Msg)
- #define `__glibcxx_requires_heap`(_First, _Last)
- #define `__glibcxx_requires_heap_pred`(_First, _Last, _Pred)
- #define `__glibcxx_requires_irreflexive`(_First, _Last)
- #define `__glibcxx_requires_irreflexive2`(_First, _Last)
- #define `__glibcxx_requires_irreflexive_pred`(_First, _Last, _Pred)
- #define `__glibcxx_requires_irreflexive_pred2`(_First, _Last, _Pred)
- #define `__glibcxx_requires_non_empty_range`(_First, _Last)
- #define `__glibcxx_requires_nonempty`()
- #define `__glibcxx_requires_partitioned_lower`(_First, _Last, _Value)
- #define `__glibcxx_requires_partitioned_lower_pred`(_First, _Last, _Value, _Pred)
- #define `__glibcxx_requires_partitioned_upper`(_First, _Last, _Value)
- #define `__glibcxx_requires_partitioned_upper_pred`(_First, _Last, _Value, _Pred)
- #define `__glibcxx_requires_sorted`(_First, _Last)
- #define `__glibcxx_requires_sorted_pred`(_First, _Last, _Pred)
- #define `__glibcxx_requires_sorted_set`(_First1, _Last1, _First2)
- #define `__glibcxx_requires_sorted_set_pred`(_First1, _Last1, _First2, _Pred)
- #define `__glibcxx_requires_string`(_String)
- #define `__glibcxx_requires_string_len`(_String, _Len)
- #define `__glibcxx_requires_subscript`(_N)
- #define `__glibcxx_requires_valid_range`(_First, _Last)

6.153.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.154 `debug_allocator.h` File Reference

Classes

- class [__gnu_cxx::debug_allocator<_Alloc>](#)

Namespaces

- [__gnu_cxx](#)

Functions

- `template<typename _Alloc>
bool __gnu_cxx::operator!= (const debug_allocator<_Alloc> &__lhs, const debug_allocator<_Alloc> &__rhs)`

6.154.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.155 `debug_fn_imps.hpp` File Reference

6.155.1 Detailed Description

Contains an implementation class for a `binary_heap`.

6.156 `debug_fn_imps.hpp` File Reference

6.156.1 Detailed Description

Contains an implementation for `binomial_heap_`.

6.157 `debug_fn_imps.hpp` File Reference

6.157.1 Detailed Description

Contains an implementation class for a base of binomial heaps.

6.158 `debug_fn_imps.hpp` File Reference

6.158.1 Detailed Description

Contains an implementation class for `bin_search_tree_`.

6.159 `debug_fn_imps.hpp` File Reference

6.159.1 Detailed Description

Contains implementations of `cc_ht_map_`'s debug-mode functions.

6.160 `debug_fn_imps.hpp` File Reference

6.160.1 Detailed Description

Contains implementations of `gp_ht_map_`'s debug-mode functions.

6.161 `debug_fn_imps.hpp` File Reference

6.161.1 Detailed Description

Contains an implementation class for `left_child_next_sibling_heap_`.

6.162 `debug_fn_imps.hpp` File Reference

6.162.1 Detailed Description

Contains implementations of `cc_ht_map_`'s debug-mode functions.

6.163 `debug_fn_imps.hpp` File Reference

6.163.1 Detailed Description

Contains an implementation class for `ov_tree_`.

6.164 `debug_fn_imps.hpp` File Reference

6.164.1 Detailed Description

Contains an implementation class for a pairing heap.

6.165 `debug_fn_imps.hpp` File Reference

6.165.1 Detailed Description

Contains an implementation class for `pat_trie_`.

6.166 `debug_fn_imps.hpp` File Reference

6.166.1 Detailed Description

Contains an implementation for `rb_tree_`.

6.167 `debug_fn_imps.hpp` File Reference

6.167.1 Detailed Description

Contains an implementation for `rc_binomial_heap_`.

6.168 `debug_fn_imps.hpp` File Reference

6.168.1 Detailed Description

Contains an implementation class for `splay_tree_`.

6.169 `debug_fn_imps.hpp` File Reference

6.169.1 Detailed Description

Contains an implementation for `thin_heap_`.

6.170 `debug_map_base.hpp` File Reference

6.170.1 Detailed Description

Contains a debug-mode base for all maps.

6.171 `debug_no_store_hash_fn_imps.hpp` File Reference

6.171.1 Detailed Description

Contains implementations of `cc_ht_map_'s` debug-mode functions.

6.172 `debug_no_store_hash_fn_imps.hpp` File Reference

6.172.1 Detailed Description

Contains implementations of `gp_ht_map_'s` debug-mode functions.

6.173 `debug_store_hash_fn_imps.hpp` File Reference

6.173.1 Detailed Description

Contains implementations of `cc_ht_map_'s` debug-mode functions.

6.174 `debug_store_hash_fn_imps.hpp` File Reference

6.174.1 Detailed Description

Contains implementations of `gp_ht_map_'s` debug-mode functions.

6.175 `decimal` File Reference

Classes

- class [std::decimal::decimal128](#)
- class [std::decimal::decimal32](#)
- class [std::decimal::decimal64](#)

Namespaces

- [std](#)
- [std::decimal](#)

Macros

- `#define _DECLARE_DECIMAL128_COMPOUND_ASSIGNMENT(_Op)`
- `#define _DECLARE_DECIMAL32_COMPOUND_ASSIGNMENT(_Op)`
- `#define _DECLARE_DECIMAL64_COMPOUND_ASSIGNMENT(_Op)`
- `#define _DECLARE_DECIMAL_BINARY_OP_WITH_DEC(_Op, _T1, _T2, _T3)`
- `#define _DECLARE_DECIMAL_BINARY_OP_WITH_INT(_Op, _Tp)`
- `#define _DECLARE_DECIMAL_COMPARISON(_Op, _Tp)`
- `#define _GLIBCXX_DECIMAL`
- `#define _GLIBCXX_USE_DECIMAL_`

Functions

- `double std::decimal::decimal128_to_double (decimal128 __d)`
- `float std::decimal::decimal128_to_float (decimal128 __d)`
- `long double std::decimal::decimal128_to_long_double (decimal128 __d)`
- `long long std::decimal::decimal128_to_long_long (decimal128 __d)`
- `double std::decimal::decimal32_to_double (decimal32 __d)`
- `float std::decimal::decimal32_to_float (decimal32 __d)`
- `long double std::decimal::decimal32_to_long_double (decimal32 __d)`
- `long long std::decimal::decimal32_to_long_long (decimal32 __d)`
- `double std::decimal::decimal64_to_double (decimal64 __d)`
- `float std::decimal::decimal64_to_float (decimal64 __d)`
- `long double std::decimal::decimal64_to_long_double (decimal64 __d)`
- `long long std::decimal::decimal64_to_long_long (decimal64 __d)`
- `double std::decimal::decimal_to_double (decimal32 __d)`
- `double std::decimal::decimal_to_double (decimal64 __d)`
- `double std::decimal::decimal_to_double (decimal128 __d)`
- `float std::decimal::decimal_to_float (decimal32 __d)`
- `float std::decimal::decimal_to_float (decimal64 __d)`
- `float std::decimal::decimal_to_float (decimal128 __d)`
- `long double std::decimal::decimal_to_long_double (decimal32 __d)`
- `long double std::decimal::decimal_to_long_double (decimal64 __d)`
- `long double std::decimal::decimal_to_long_double (decimal128 __d)`
- `long long std::decimal::decimal_to_long_long (decimal32 __d)`
- `long long std::decimal::decimal_to_long_long (decimal64 __d)`
- `long long std::decimal::decimal_to_long_long (decimal128 __d)`
- `static decimal128 std::decimal::make_decimal128 (long long __coeff, int __exp)`
- `static decimal128 std::decimal::make_decimal128 (unsigned long long __coeff, int __exp)`
- `static decimal32 std::decimal::make_decimal32 (long long __coeff, int __exp)`
- `static decimal32 std::decimal::make_decimal32 (unsigned long long __coeff, int __exp)`
- `static decimal64 std::decimal::make_decimal64 (long long __coeff, int __exp)`
- `static decimal64 std::decimal::make_decimal64 (unsigned long long __coeff, int __exp)`
- `bool std::decimal::operator!= (decimal32 __lhs, decimal32 __rhs)`

- decimal32 **std::decimal::operator*** (int __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator*** (unsigned int __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator*** (long __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator*** (unsigned long __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator*** (long long __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator*** (unsigned long long __lhs, decimal32 __rhs)
- decimal64 **std::decimal::operator*** (decimal32 __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator*** (decimal64 __lhs, decimal32 __rhs)
- decimal64 **std::decimal::operator*** (decimal64 __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator*** (decimal64 __lhs, int __rhs)
- decimal64 **std::decimal::operator*** (decimal64 __lhs, unsigned int __rhs)
- decimal64 **std::decimal::operator*** (decimal64 __lhs, long __rhs)
- decimal64 **std::decimal::operator*** (decimal64 __lhs, unsigned long __rhs)
- decimal64 **std::decimal::operator*** (decimal64 __lhs, long long __rhs)
- decimal64 **std::decimal::operator*** (decimal64 __lhs, unsigned long long __rhs)
- decimal64 **std::decimal::operator*** (int __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator*** (unsigned int __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator*** (long __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator*** (unsigned long __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator*** (long long __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator*** (unsigned long long __lhs, decimal64 __rhs)
- decimal128 **std::decimal::operator*** (decimal32 __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator*** (decimal64 __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator*** (decimal128 __lhs, decimal32 __rhs)
- decimal128 **std::decimal::operator*** (decimal128 __lhs, decimal64 __rhs)
- decimal128 **std::decimal::operator*** (decimal128 __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator*** (decimal128 __lhs, int __rhs)
- decimal128 **std::decimal::operator*** (decimal128 __lhs, unsigned int __rhs)
- decimal128 **std::decimal::operator*** (decimal128 __lhs, long __rhs)
- decimal128 **std::decimal::operator*** (decimal128 __lhs, unsigned long __rhs)
- decimal128 **std::decimal::operator*** (decimal128 __lhs, long long __rhs)
- decimal128 **std::decimal::operator*** (decimal128 __lhs, unsigned long long __rhs)
- decimal128 **std::decimal::operator*** (int __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator*** (unsigned int __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator*** (long __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator*** (unsigned long __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator*** (long long __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator*** (unsigned long long __lhs, decimal128 __rhs)
- decimal32 **std::decimal::operator+** (decimal32 __rhs)
- decimal64 **std::decimal::operator+** (decimal64 __rhs)
- decimal128 **std::decimal::operator+** (decimal128 __rhs)
- decimal32 **std::decimal::operator+** (decimal32 __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator+** (decimal32 __lhs, int __rhs)
- decimal32 **std::decimal::operator+** (decimal32 __lhs, unsigned int __rhs)
- decimal32 **std::decimal::operator+** (decimal32 __lhs, long __rhs)
- decimal32 **std::decimal::operator+** (decimal32 __lhs, unsigned long __rhs)
- decimal32 **std::decimal::operator+** (decimal32 __lhs, long long __rhs)
- decimal32 **std::decimal::operator+** (decimal32 __lhs, unsigned long long __rhs)
- decimal32 **std::decimal::operator+** (int __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator+** (unsigned int __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator+** (long __lhs, decimal32 __rhs)

- decimal32 **std::decimal::operator+** (unsigned long __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator+** (long long __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator+** (unsigned long long __lhs, decimal32 __rhs)
- decimal64 **std::decimal::operator+** (decimal32 __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator+** (decimal64 __lhs, decimal32 __rhs)
- decimal64 **std::decimal::operator+** (decimal64 __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator+** (unsigned long long __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator+** (decimal64 __lhs, int __rhs)
- decimal64 **std::decimal::operator+** (decimal64 __lhs, unsigned int __rhs)
- decimal64 **std::decimal::operator+** (decimal64 __lhs, long __rhs)
- decimal64 **std::decimal::operator+** (decimal64 __lhs, unsigned long __rhs)
- decimal64 **std::decimal::operator+** (decimal64 __lhs, long long __rhs)
- decimal64 **std::decimal::operator+** (decimal64 __lhs, unsigned long long __rhs)
- decimal64 **std::decimal::operator+** (int __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator+** (unsigned int __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator+** (long __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator+** (unsigned long __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator+** (long long __lhs, decimal64 __rhs)
- decimal128 **std::decimal::operator+** (decimal32 __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator+** (decimal64 __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator+** (decimal128 __lhs, decimal32 __rhs)
- decimal128 **std::decimal::operator+** (decimal128 __lhs, decimal64 __rhs)
- decimal128 **std::decimal::operator+** (decimal128 __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator+** (decimal128 __lhs, int __rhs)
- decimal128 **std::decimal::operator+** (decimal128 __lhs, unsigned int __rhs)
- decimal128 **std::decimal::operator+** (decimal128 __lhs, long __rhs)
- decimal128 **std::decimal::operator+** (decimal128 __lhs, unsigned long __rhs)
- decimal128 **std::decimal::operator+** (decimal128 __lhs, long long __rhs)
- decimal128 **std::decimal::operator+** (decimal128 __lhs, unsigned long long __rhs)
- decimal128 **std::decimal::operator+** (int __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator+** (unsigned int __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator+** (long __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator+** (unsigned long __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator+** (long long __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator+** (unsigned long long __lhs, decimal128 __rhs)
- decimal32 **std::decimal::operator-** (decimal32 __rhs)
- decimal64 **std::decimal::operator-** (decimal64 __rhs)
- decimal128 **std::decimal::operator-** (decimal128 __rhs)
- decimal32 **std::decimal::operator-** (decimal32 __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator-** (decimal32 __lhs, int __rhs)
- decimal32 **std::decimal::operator-** (decimal32 __lhs, unsigned int __rhs)
- decimal32 **std::decimal::operator-** (decimal32 __lhs, long __rhs)
- decimal32 **std::decimal::operator-** (decimal32 __lhs, unsigned long __rhs)
- decimal32 **std::decimal::operator-** (decimal32 __lhs, long long __rhs)
- decimal32 **std::decimal::operator-** (decimal32 __lhs, unsigned long long __rhs)
- decimal32 **std::decimal::operator-** (int __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator-** (unsigned int __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator-** (long __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator-** (unsigned long __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator-** (long long __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator-** (unsigned long long __lhs, decimal32 __rhs)

- decimal64 **std::decimal::operator-** (decimal32 __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator-** (decimal64 __lhs, decimal32 __rhs)
- decimal64 **std::decimal::operator-** (decimal64 __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator-** (decimal64 __lhs, int __rhs)
- decimal64 **std::decimal::operator-** (decimal64 __lhs, unsigned int __rhs)
- decimal64 **std::decimal::operator-** (decimal64 __lhs, long __rhs)
- decimal64 **std::decimal::operator-** (decimal64 __lhs, unsigned long __rhs)
- decimal64 **std::decimal::operator-** (decimal64 __lhs, long long __rhs)
- decimal64 **std::decimal::operator-** (decimal64 __lhs, unsigned long long __rhs)
- decimal64 **std::decimal::operator-** (int __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator-** (unsigned int __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator-** (long __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator-** (unsigned long __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator-** (long long __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator-** (unsigned long long __lhs, decimal64 __rhs)
- decimal128 **std::decimal::operator-** (decimal32 __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator-** (decimal64 __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator-** (decimal128 __lhs, decimal32 __rhs)
- decimal128 **std::decimal::operator-** (decimal128 __lhs, decimal64 __rhs)
- decimal128 **std::decimal::operator-** (decimal128 __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator-** (decimal128 __lhs, int __rhs)
- decimal128 **std::decimal::operator-** (decimal128 __lhs, unsigned int __rhs)
- decimal128 **std::decimal::operator-** (decimal128 __lhs, long __rhs)
- decimal128 **std::decimal::operator-** (decimal128 __lhs, unsigned long __rhs)
- decimal128 **std::decimal::operator-** (decimal128 __lhs, long long __rhs)
- decimal128 **std::decimal::operator-** (decimal128 __lhs, unsigned long long __rhs)
- decimal128 **std::decimal::operator-** (int __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator-** (unsigned int __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator-** (long __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator-** (unsigned long __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator-** (long long __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator-** (unsigned long long __lhs, decimal128 __rhs)
- decimal32 **std::decimal::operator/** (decimal32 __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator/** (decimal32 __lhs, int __rhs)
- decimal32 **std::decimal::operator/** (decimal32 __lhs, unsigned int __rhs)
- decimal32 **std::decimal::operator/** (decimal32 __lhs, long __rhs)
- decimal32 **std::decimal::operator/** (decimal32 __lhs, unsigned long __rhs)
- decimal32 **std::decimal::operator/** (decimal32 __lhs, long long __rhs)
- decimal32 **std::decimal::operator/** (decimal32 __lhs, unsigned long long __rhs)
- decimal32 **std::decimal::operator/** (int __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator/** (unsigned int __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator/** (long __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator/** (unsigned long __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator/** (long long __lhs, decimal32 __rhs)
- decimal32 **std::decimal::operator/** (unsigned long long __lhs, decimal32 __rhs)
- decimal64 **std::decimal::operator/** (decimal32 __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator/** (decimal64 __lhs, decimal32 __rhs)
- decimal64 **std::decimal::operator/** (decimal64 __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator/** (decimal64 __lhs, int __rhs)
- decimal64 **std::decimal::operator/** (decimal64 __lhs, unsigned int __rhs)
- decimal64 **std::decimal::operator/** (decimal64 __lhs, long __rhs)

- decimal64 **std::decimal::operator/** (decimal64 __lhs, unsigned long __rhs)
- decimal64 **std::decimal::operator/** (decimal64 __lhs, long long __rhs)
- decimal64 **std::decimal::operator/** (decimal64 __lhs, unsigned long long __rhs)
- decimal64 **std::decimal::operator/** (int __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator/** (unsigned int __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator/** (long __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator/** (unsigned long __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator/** (long long __lhs, decimal64 __rhs)
- decimal64 **std::decimal::operator/** (unsigned long long __lhs, decimal64 __rhs)
- decimal128 **std::decimal::operator/** (decimal32 __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator/** (decimal64 __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator/** (decimal128 __lhs, decimal32 __rhs)
- decimal128 **std::decimal::operator/** (decimal128 __lhs, decimal64 __rhs)
- decimal128 **std::decimal::operator/** (decimal128 __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator/** (decimal128 __lhs, long __rhs)
- decimal128 **std::decimal::operator/** (long long __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator/** (decimal128 __lhs, int __rhs)
- decimal128 **std::decimal::operator/** (decimal128 __lhs, unsigned int __rhs)
- decimal128 **std::decimal::operator/** (decimal128 __lhs, unsigned long __rhs)
- decimal128 **std::decimal::operator/** (decimal128 __lhs, long long __rhs)
- decimal128 **std::decimal::operator/** (decimal128 __lhs, unsigned long long __rhs)
- decimal128 **std::decimal::operator/** (int __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator/** (unsigned int __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator/** (long __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator/** (unsigned long __lhs, decimal128 __rhs)
- decimal128 **std::decimal::operator/** (unsigned long long __lhs, decimal128 __rhs)
- bool **std::decimal::operator<** (unsigned long __lhs, decimal32 __rhs)
- bool **std::decimal::operator<** (decimal32 __lhs, decimal32 __rhs)
- bool **std::decimal::operator<** (decimal32 __lhs, decimal64 __rhs)
- bool **std::decimal::operator<** (decimal32 __lhs, decimal128 __rhs)
- bool **std::decimal::operator<** (decimal32 __lhs, int __rhs)
- bool **std::decimal::operator<** (decimal32 __lhs, long __rhs)
- bool **std::decimal::operator<** (decimal32 __lhs, unsigned long __rhs)
- bool **std::decimal::operator<** (decimal32 __lhs, long long __rhs)
- bool **std::decimal::operator<** (int __lhs, decimal32 __rhs)
- bool **std::decimal::operator<** (long __lhs, decimal32 __rhs)
- bool **std::decimal::operator<** (decimal32 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator<** (long long __lhs, decimal32 __rhs)
- bool **std::decimal::operator<** (unsigned long long __lhs, decimal32 __rhs)
- bool **std::decimal::operator<** (unsigned int __lhs, decimal32 __rhs)
- bool **std::decimal::operator<** (decimal32 __lhs, unsigned int __rhs)
- bool **std::decimal::operator<** (long __lhs, decimal64 __rhs)
- bool **std::decimal::operator<** (unsigned long __lhs, decimal64 __rhs)
- bool **std::decimal::operator<** (decimal64 __lhs, decimal64 __rhs)
- bool **std::decimal::operator<** (unsigned long long __lhs, decimal64 __rhs)
- bool **std::decimal::operator<** (long long __lhs, decimal64 __rhs)
- bool **std::decimal::operator<** (decimal64 __lhs, decimal32 __rhs)
- bool **std::decimal::operator<** (decimal64 __lhs, decimal128 __rhs)
- bool **std::decimal::operator<** (decimal64 __lhs, unsigned int __rhs)
- bool **std::decimal::operator<** (decimal64 __lhs, int __rhs)
- bool **std::decimal::operator<** (int __lhs, decimal64 __rhs)

- bool **std::decimal::operator<** (unsigned int __lhs, decimal64 __rhs)
- bool **std::decimal::operator<** (decimal64 __lhs, long long __rhs)
- bool **std::decimal::operator<** (decimal64 __lhs, long __rhs)
- bool **std::decimal::operator<** (decimal64 __lhs, unsigned long __rhs)
- bool **std::decimal::operator<** (decimal64 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator<** (unsigned long __lhs, decimal128 __rhs)
- bool **std::decimal::operator<** (decimal128 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator<** (decimal128 __lhs, unsigned int __rhs)
- bool **std::decimal::operator<** (unsigned long long __lhs, decimal128 __rhs)
- bool **std::decimal::operator<** (decimal128 __lhs, decimal32 __rhs)
- bool **std::decimal::operator<** (int __lhs, decimal128 __rhs)
- bool **std::decimal::operator<** (unsigned int __lhs, decimal128 __rhs)
- bool **std::decimal::operator<** (long long __lhs, decimal128 __rhs)
- bool **std::decimal::operator<** (long __lhs, decimal128 __rhs)
- bool **std::decimal::operator<** (decimal128 __lhs, unsigned long __rhs)
- bool **std::decimal::operator<** (decimal128 __lhs, int __rhs)
- bool **std::decimal::operator<** (decimal128 __lhs, decimal64 __rhs)
- bool **std::decimal::operator<** (decimal128 __lhs, long __rhs)
- bool **std::decimal::operator<** (decimal128 __lhs, decimal128 __rhs)
- bool **std::decimal::operator<** (decimal128 __lhs, long long __rhs)
- bool **std::decimal::operator==** (decimal32 __lhs, unsigned long __rhs)
- bool **std::decimal::operator==** (decimal32 __lhs, decimal128 __rhs)
- bool **std::decimal::operator==** (decimal32 __lhs, decimal32 __rhs)
- bool **std::decimal::operator==** (decimal32 __lhs, decimal64 __rhs)
- bool **std::decimal::operator==** (decimal32 __lhs, int __rhs)
- bool **std::decimal::operator==** (decimal32 __lhs, unsigned int __rhs)
- bool **std::decimal::operator==** (decimal32 __lhs, long __rhs)
- bool **std::decimal::operator==** (decimal32 __lhs, long long __rhs)
- bool **std::decimal::operator==** (decimal32 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator==** (int __lhs, decimal32 __rhs)
- bool **std::decimal::operator==** (unsigned int __lhs, decimal32 __rhs)
- bool **std::decimal::operator==** (long __lhs, decimal32 __rhs)
- bool **std::decimal::operator==** (unsigned long __lhs, decimal32 __rhs)
- bool **std::decimal::operator==** (long long __lhs, decimal32 __rhs)
- bool **std::decimal::operator==** (unsigned long long __lhs, decimal32 __rhs)
- bool **std::decimal::operator==** (unsigned long long __lhs, decimal64 __rhs)
- bool **std::decimal::operator==** (long __lhs, decimal64 __rhs)
- bool **std::decimal::operator==** (decimal64 __lhs, long long __rhs)
- bool **std::decimal::operator==** (decimal64 __lhs, unsigned int __rhs)
- bool **std::decimal::operator==** (decimal64 __lhs, decimal128 __rhs)
- bool **std::decimal::operator==** (long long __lhs, decimal64 __rhs)
- bool **std::decimal::operator==** (decimal64 __lhs, int __rhs)
- bool **std::decimal::operator==** (decimal64 __lhs, long __rhs)
- bool **std::decimal::operator==** (decimal64 __lhs, decimal32 __rhs)
- bool **std::decimal::operator==** (decimal64 __lhs, decimal64 __rhs)
- bool **std::decimal::operator==** (decimal64 __lhs, unsigned long __rhs)
- bool **std::decimal::operator==** (decimal64 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator==** (int __lhs, decimal64 __rhs)
- bool **std::decimal::operator==** (unsigned int __lhs, decimal64 __rhs)
- bool **std::decimal::operator==** (unsigned long __lhs, decimal64 __rhs)
- bool **std::decimal::operator==** (int __lhs, decimal128 __rhs)

- bool **std::decimal::operator==** (unsigned int __lhs, decimal128 __rhs)
- bool **std::decimal::operator==** (long __lhs, decimal128 __rhs)
- bool **std::decimal::operator==** (long long __lhs, decimal128 __rhs)
- bool **std::decimal::operator==** (unsigned long long __lhs, decimal128 __rhs)
- bool **std::decimal::operator==** (unsigned long __lhs, decimal128 __rhs)
- bool **std::decimal::operator==** (decimal128 __lhs, decimal32 __rhs)
- bool **std::decimal::operator==** (decimal128 __lhs, unsigned int __rhs)
- bool **std::decimal::operator==** (decimal128 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator==** (decimal128 __lhs, unsigned long __rhs)
- bool **std::decimal::operator==** (decimal128 __lhs, decimal128 __rhs)
- bool **std::decimal::operator==** (decimal128 __lhs, long long __rhs)
- bool **std::decimal::operator==** (decimal128 __lhs, decimal64 __rhs)
- bool **std::decimal::operator==** (decimal128 __lhs, int __rhs)
- bool **std::decimal::operator==** (decimal128 __lhs, long __rhs)
- bool **std::decimal::operator>** (unsigned int __lhs, decimal32 __rhs)
- bool **std::decimal::operator>** (long __lhs, decimal32 __rhs)
- bool **std::decimal::operator>** (decimal32 __lhs, decimal128 __rhs)
- bool **std::decimal::operator>** (decimal32 __lhs, long long __rhs)
- bool **std::decimal::operator>** (decimal32 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator>** (decimal32 __lhs, unsigned long __rhs)
- bool **std::decimal::operator>** (decimal32 __lhs, decimal32 __rhs)
- bool **std::decimal::operator>** (decimal32 __lhs, decimal64 __rhs)
- bool **std::decimal::operator>** (decimal32 __lhs, long __rhs)
- bool **std::decimal::operator>** (unsigned long __lhs, decimal32 __rhs)
- bool **std::decimal::operator>** (unsigned long long __lhs, decimal32 __rhs)
- bool **std::decimal::operator>** (long long __lhs, decimal32 __rhs)
- bool **std::decimal::operator>** (decimal32 __lhs, unsigned int __rhs)
- bool **std::decimal::operator>** (int __lhs, decimal32 __rhs)
- bool **std::decimal::operator>** (decimal32 __lhs, int __rhs)
- bool **std::decimal::operator>** (decimal64 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator>** (decimal64 __lhs, decimal32 __rhs)
- bool **std::decimal::operator>** (unsigned long __lhs, decimal64 __rhs)
- bool **std::decimal::operator>** (unsigned long long __lhs, decimal64 __rhs)
- bool **std::decimal::operator>** (long long __lhs, decimal64 __rhs)
- bool **std::decimal::operator>** (int __lhs, decimal64 __rhs)
- bool **std::decimal::operator>** (decimal64 __lhs, unsigned int __rhs)
- bool **std::decimal::operator>** (decimal64 __lhs, unsigned long __rhs)
- bool **std::decimal::operator>** (decimal64 __lhs, decimal128 __rhs)
- bool **std::decimal::operator>** (decimal64 __lhs, long __rhs)
- bool **std::decimal::operator>** (decimal64 __lhs, long long __rhs)
- bool **std::decimal::operator>** (decimal64 __lhs, decimal64 __rhs)
- bool **std::decimal::operator>** (decimal64 __lhs, int __rhs)
- bool **std::decimal::operator>** (long __lhs, decimal64 __rhs)
- bool **std::decimal::operator>** (unsigned int __lhs, decimal64 __rhs)
- bool **std::decimal::operator>** (decimal128 __lhs, decimal128 __rhs)
- bool **std::decimal::operator>** (int __lhs, decimal128 __rhs)
- bool **std::decimal::operator>** (decimal128 __lhs, unsigned long __rhs)
- bool **std::decimal::operator>** (unsigned long long __lhs, decimal128 __rhs)
- bool **std::decimal::operator>** (decimal128 __lhs, unsigned int __rhs)
- bool **std::decimal::operator>** (unsigned int __lhs, decimal128 __rhs)
- bool **std::decimal::operator>** (decimal128 __lhs, decimal32 __rhs)

- bool **std::decimal::operator>** (decimal128 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator>** (decimal128 __lhs, long __rhs)
- bool **std::decimal::operator>** (unsigned long __lhs, decimal128 __rhs)
- bool **std::decimal::operator>** (decimal128 __lhs, int __rhs)
- bool **std::decimal::operator>** (long long __lhs, decimal128 __rhs)
- bool **std::decimal::operator>** (long __lhs, decimal128 __rhs)
- bool **std::decimal::operator>** (decimal128 __lhs, decimal64 __rhs)
- bool **std::decimal::operator>** (decimal128 __lhs, long long __rhs)
- bool **std::decimal::operator>=** (long long __lhs, decimal32 __rhs)
- bool **std::decimal::operator>=** (unsigned long __lhs, decimal32 __rhs)
- bool **std::decimal::operator>=** (decimal32 __lhs, decimal64 __rhs)
- bool **std::decimal::operator>=** (decimal32 __lhs, unsigned int __rhs)
- bool **std::decimal::operator>=** (decimal32 __lhs, decimal32 __rhs)
- bool **std::decimal::operator>=** (decimal32 __lhs, int __rhs)
- bool **std::decimal::operator>=** (decimal32 __lhs, decimal128 __rhs)
- bool **std::decimal::operator>=** (unsigned long long __lhs, decimal32 __rhs)
- bool **std::decimal::operator>=** (unsigned int __lhs, decimal32 __rhs)
- bool **std::decimal::operator>=** (long __lhs, decimal32 __rhs)
- bool **std::decimal::operator>=** (decimal32 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator>=** (decimal32 __lhs, long long __rhs)
- bool **std::decimal::operator>=** (int __lhs, decimal32 __rhs)
- bool **std::decimal::operator>=** (decimal32 __lhs, long __rhs)
- bool **std::decimal::operator>=** (decimal32 __lhs, unsigned long __rhs)
- bool **std::decimal::operator>=** (unsigned long long __lhs, decimal64 __rhs)
- bool **std::decimal::operator>=** (decimal64 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator>=** (decimal64 __lhs, long long __rhs)
- bool **std::decimal::operator>=** (decimal64 __lhs, decimal64 __rhs)
- bool **std::decimal::operator>=** (decimal64 __lhs, decimal32 __rhs)
- bool **std::decimal::operator>=** (decimal64 __lhs, unsigned int __rhs)
- bool **std::decimal::operator>=** (decimal64 __lhs, unsigned long __rhs)
- bool **std::decimal::operator>=** (decimal64 __lhs, decimal128 __rhs)
- bool **std::decimal::operator>=** (long __lhs, decimal64 __rhs)
- bool **std::decimal::operator>=** (decimal64 __lhs, long __rhs)
- bool **std::decimal::operator>=** (unsigned int __lhs, decimal64 __rhs)
- bool **std::decimal::operator>=** (decimal64 __lhs, int __rhs)
- bool **std::decimal::operator>=** (unsigned long __lhs, decimal64 __rhs)
- bool **std::decimal::operator>=** (int __lhs, decimal64 __rhs)
- bool **std::decimal::operator>=** (long long __lhs, decimal64 __rhs)
- bool **std::decimal::operator>=** (decimal128 __lhs, int __rhs)
- bool **std::decimal::operator>=** (int __lhs, decimal128 __rhs)
- bool **std::decimal::operator>=** (decimal128 __lhs, unsigned long __rhs)
- bool **std::decimal::operator>=** (long long __lhs, decimal128 __rhs)
- bool **std::decimal::operator>=** (decimal128 __lhs, decimal64 __rhs)
- bool **std::decimal::operator>=** (unsigned long __lhs, decimal128 __rhs)
- bool **std::decimal::operator>=** (decimal128 __lhs, decimal32 __rhs)
- bool **std::decimal::operator>=** (decimal128 __lhs, long __rhs)
- bool **std::decimal::operator>=** (decimal128 __lhs, unsigned int __rhs)
- bool **std::decimal::operator>=** (decimal128 __lhs, long long __rhs)
- bool **std::decimal::operator>=** (decimal128 __lhs, decimal128 __rhs)
- bool **std::decimal::operator>=** (unsigned int __lhs, decimal128 __rhs)
- bool **std::decimal::operator>=** (decimal128 __lhs, unsigned long long __rhs)
- bool **std::decimal::operator>=** (long __lhs, decimal128 __rhs)
- bool **std::decimal::operator>=** (unsigned long long __lhs, decimal128 __rhs)

6.175.1 Detailed Description

This is a Standard C++ Library header.

6.176 deque File Reference

Macros

- `#define _GLIBCXX_DEQUE`

6.176.1 Detailed Description

This is a Standard C++ Library header.

6.177 deque File Reference

Classes

- class `std::__debug::deque< _Tp, _Allocator >`

Namespaces

- `std`
- `std::__debug`

Macros

- `#define _GLIBCXX_DEBUG_DEQUE`

Functions

- `template<typename _Tp, typename _Alloc >`
`bool std::__debug::operator!= (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool std::__debug::operator< (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool std::__debug::operator<= (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool std::__debug::operator== (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool std::__debug::operator> (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool std::__debug::operator>= (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`void std::__debug::swap (deque< _Tp, _Alloc > &__lhs, deque< _Tp, _Alloc > &__rhs) noexcept(/*conditional */)`

6.177.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.178 deque File Reference

Classes

- class [std::__profile::deque<_Tp, _Allocator >](#)

Namespaces

- [std](#)
- [std::__profile](#)

Macros

- `#define _GLIBCXX_PROFILE_DEQUE`

Functions

- `template<typename _Tp, typename _Alloc >
bool std::__profile::operator!= (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool std::__profile::operator< (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool std::__profile::operator<= (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool std::__profile::operator== (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool std::__profile::operator> (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool std::__profile::operator>= (const deque< _Tp, _Alloc > &__lhs, const deque< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
void std::__profile::swap (deque< _Tp, _Alloc > &__lhs, deque< _Tp, _Alloc > &__rhs) noexcept(/*conditional */)`

6.178.1 Detailed Description

This file is a GNU profile extension to the Standard C++ Library.

6.179 deque File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_DEQUE`

Typedefs

- `template<typename _Tp >`
using **`std::experimental::fundamentals_v2::pmr::deque`** = `std::deque`< _Tp, `polymorphic_allocator`< _Tp >>

Functions

- `template<typename _Tp, typename _Alloc, typename _Up >`
void **`std::experimental::fundamentals_v2::erase`** (`deque`< _Tp, _Alloc > &__cont, const _Up &__value)
- `template<typename _Tp, typename _Alloc, typename _Predicate >`
void **`std::experimental::fundamentals_v2::erase_if`** (`deque`< _Tp, _Alloc > &__cont, _Predicate __pred)

6.179.1 Detailed Description

This is a TS C++ Library header.

6.180 deque.tcc File Reference

Namespaces

- `std`

Macros

- `#define _DEQUE_TCC`

Functions

- `template<typename _Tp >`
`_Deque_iterator`< _Tp, _Tp &, _Tp * > **`std::copy`** (`_Deque_iterator`< _Tp, const _Tp &, const _Tp * > __first, `_Deque_iterator`< _Tp, const _Tp &, const _Tp * > __last, `_Deque_iterator`< _Tp, _Tp &, _Tp * > __result)
- `template<typename _Tp >`
`_Deque_iterator`< _Tp, _Tp &, _Tp * > **`std::copy_backward`** (`_Deque_iterator`< _Tp, const _Tp &, const _Tp * > __first, `_Deque_iterator`< _Tp, const _Tp &, const _Tp * > __last, `_Deque_iterator`< _Tp, _Tp &, _Tp * > __result)
- `template<typename _Tp >`
void **`std::fill`** (const `_Deque_iterator`< _Tp, _Tp &, _Tp * > &__first, const `_Deque_iterator`< _Tp, _Tp &, _Tp * > &__last, const _Tp &__value)
- `template<typename _Tp >`
`_Deque_iterator`< _Tp, _Tp &, _Tp * > **`std::move`** (`_Deque_iterator`< _Tp, const _Tp &, const _Tp * > __first, `_Deque_iterator`< _Tp, const _Tp &, const _Tp * > __last, `_Deque_iterator`< _Tp, _Tp &, _Tp * > __result)
- `template<typename _Tp >`
`_Deque_iterator`< _Tp, _Tp &, _Tp * > **`std::move_backward`** (`_Deque_iterator`< _Tp, const _Tp &, const _Tp * > __first, `_Deque_iterator`< _Tp, const _Tp &, const _Tp * > __last, `_Deque_iterator`< _Tp, _Tp &, _Tp * > __result)

6.180.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<deque>`.

6.181 `direct_mask_range_hashing_imp.hpp` File Reference

6.181.1 Detailed Description

Contains a range-hashing policy implementation

6.182 `direct_mod_range_hashing_imp.hpp` File Reference

6.182.1 Detailed Description

Contains a range-hashing policy implementation

6.183 `dynamic_bitset` File Reference

Classes

- struct `std::tr2::__dynamic_bitset_base<_WordT, _Alloc>`
- class `std::tr2::dynamic_bitset<_WordT, _Alloc>`
- class `std::tr2::dynamic_bitset<_WordT, _Alloc>::reference`

Namespaces

- `std`
- `std::tr2`

Macros

- `#define _GLIBCXX_TR2_DYNAMIC_BITSET`

Functions

- `template<typename _CharT, typename _Traits, typename _WordT, typename _Alloc >`
`std::basic_ostream< _CharT, _Traits > & std::tr2::operator<< (std::basic_ostream< _CharT, _Traits > &__os,`
`const dynamic_bitset< _WordT, _Alloc > &__x)`
- `template<typename _WordT, typename _Alloc >`
`bool std::tr2::operator!= (const dynamic_bitset< _WordT, _Alloc > &__lhs, const dynamic_bitset< _WordT, _Alloc`
`> &__rhs)`
- `template<typename _WordT, typename _Alloc >`
`bool std::tr2::operator<= (const dynamic_bitset< _WordT, _Alloc > &__lhs, const dynamic_bitset< _WordT, _Alloc`
`> &__rhs)`
- `template<typename _WordT, typename _Alloc >`
`bool std::tr2::operator> (const dynamic_bitset< _WordT, _Alloc > &__lhs, const dynamic_bitset< _WordT, _Alloc`
`> &__rhs)`
- `template<typename _WordT, typename _Alloc >`
`bool std::tr2::operator>= (const dynamic_bitset< _WordT, _Alloc > &__lhs, const dynamic_bitset< _WordT, _Alloc`
`> &__rhs)`
- `template<typename _WordT, typename _Alloc >`
`dynamic_bitset< _WordT, _Alloc > std::tr2::operator& (const dynamic_bitset< _WordT, _Alloc > &__x, const`
`dynamic_bitset< _WordT, _Alloc > &__y)`
- `template<typename _WordT, typename _Alloc >`
`dynamic_bitset< _WordT, _Alloc > std::tr2::operator| (const dynamic_bitset< _WordT, _Alloc > &__x, const`
`dynamic_bitset< _WordT, _Alloc > &__y)`
- `template<typename _WordT, typename _Alloc >`
`dynamic_bitset< _WordT, _Alloc > std::tr2::operator^ (const dynamic_bitset< _WordT, _Alloc > &__x, const`
`dynamic_bitset< _WordT, _Alloc > &__y)`
- `template<typename _WordT, typename _Alloc >`
`dynamic_bitset< _WordT, _Alloc > std::tr2::operator- (const dynamic_bitset< _WordT, _Alloc > &__x, const`
`dynamic_bitset< _WordT, _Alloc > &__y)`

6.183.1 Detailed Description

This is a TR2 C++ Library header.

6.184 dynamic_bitset.tcc File Reference

Namespaces

- `std`
- `std::tr2`

Macros

- `#define _GLIBCXX_TR2_DYNAMIC_BITSET_TCC`

Functions

- `template<typename _CharT, typename _Traits, typename _WordT, typename _Alloc >
std::basic_istream< _CharT, _Traits > & std::tr2::operator>> (std::basic_istream< _CharT, _Traits > &__x
is, dynamic_bitset< _WordT, _Alloc > &__x)`

6.184.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<tr2/dynamic_bitset>`.

6.185 enable_special_members.h File Reference

Classes

- `struct std::_Enable_copy_move< _Copy, _CopyAssignment, _Move, _MoveAssignment, _Tag >`
- `struct std::_Enable_default_constructor< _Switch, _Tag >`
- `struct std::_Enable_destructor< _Switch, _Tag >`
- `struct std::_Enable_special_members< _Default, _Destructor, _Copy, _CopyAssignment, _Move, _MoveAssignment, _Tag >`

Namespaces

- `std`

6.185.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly.

6.186 enc_filebuf.h File Reference

Classes

- `class __gnu_cxx::enc_filebuf< _CharT >`

Namespaces

- `__gnu_cxx`

6.186.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.187 `entry_cmp.hpp` File Reference

Classes

- struct [__gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, No_Throw >](#)
- struct [__gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, false >](#)
- struct [__gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, false >::type](#)
- struct [__gnu_pbds::detail::entry_cmp<_VTp, Cmp_Fn, _Alloc, true >](#)

Namespaces

- [__gnu_pbds](#)

6.187.1 Detailed Description

Contains an implementation class for a `binary_heap`.

6.188 `entry_list_fn_imps.hpp` File Reference

6.188.1 Detailed Description

Contains implementations of `cc_ht_map_`'s entry-list related functions.

6.189 `entry_metadata_base.hpp` File Reference

Namespaces

- [__gnu_pbds](#)

6.189.1 Detailed Description

Contains an implementation for a list update map.

6.190 `entry_pred.hpp` File Reference

Classes

- struct [__gnu_pbds::detail::entry_pred<_VTp, Pred, _Alloc, No_Throw >](#)
- struct [__gnu_pbds::detail::entry_pred<_VTp, Pred, _Alloc, false >](#)
- struct [__gnu_pbds::detail::entry_pred<_VTp, Pred, _Alloc, true >](#)

Namespaces

- [__gnu_pbds](#)

6.190.1 Detailed Description

Contains an implementation class for a `binary_heap`.

6.191 `eq_by_less.hpp` File Reference

Classes

- struct [__gnu_pbds::detail::eq_by_less< Key, Cmp_Fn >](#)

Namespaces

- [__gnu_pbds](#)

6.191.1 Detailed Description

Contains an equivalence function.

6.192 `equally_split.h` File Reference

Namespaces

- [__gnu_parallel](#)

Functions

- template<typename `_DifferenceType` , typename `_OutputIterator` >
`_OutputIterator` [__gnu_parallel::__equally_split](#) (`_DifferenceType` `__n`, `_ThreadIndex` `__num_threads`, `_OutputIterator` `__s`)
- template<typename `_DifferenceType` >
`_DifferenceType` [__gnu_parallel::__equally_split_point](#) (`_DifferenceType` `__n`, `_ThreadIndex` `__num_threads`, `_ThreadIndex` `__thread_no`)

6.192.1 Detailed Description

This file is a GNU parallel extension to the Standard C++ Library.

6.193 erase_fn_imps.hpp File Reference**6.193.1 Detailed Description**

Contains an implementation class for a binary_heap.

6.194 erase_fn_imps.hpp File Reference**6.194.1 Detailed Description**

Contains an implementation class for a base of binomial heaps.

6.195 erase_fn_imps.hpp File Reference**6.195.1 Detailed Description**

Contains an implementation class for bin_search_tree_.

6.196 erase_fn_imps.hpp File Reference**6.196.1 Detailed Description**

Contains implementations of cc_ht_map_'s erase related functions.

6.197 erase_fn_imps.hpp File Reference**6.197.1 Detailed Description**

Contains implementations of gp_ht_map_'s erase related functions.

6.198 erase_fn_imps.hpp File Reference**6.198.1 Detailed Description**

Contains an implementation class for left_child_next_sibling_heap_.

6.199 erase_fn_imps.hpp File Reference

6.199.1 Detailed Description

Contains implementations of lu_map_.

6.200 erase_fn_imps.hpp File Reference

6.200.1 Detailed Description

Contains an implementation class for ov_tree_.

6.201 erase_fn_imps.hpp File Reference

6.201.1 Detailed Description

Contains an implementation class for a pairing heap.

6.202 erase_fn_imps.hpp File Reference

6.202.1 Detailed Description

Contains an implementation class for pat_trie.

6.203 erase_fn_imps.hpp File Reference

6.203.1 Detailed Description

Contains an implementation for rb_tree_.

6.204 erase_fn_imps.hpp File Reference

6.204.1 Detailed Description

Contains an implementation for rc_binomial_heap_.

6.205 `erase_fn_imps.hpp` File Reference

6.205.1 Detailed Description

Contains an implementation class for `splay_tree_`.

6.206 `erase_fn_imps.hpp` File Reference

6.206.1 Detailed Description

Contains an implementation for `thin_heap_`.

6.207 `erase_if.h` File Reference

Namespaces

- [std](#)

Functions

- `template<typename _Container , typename _Predicate >
void std::experimental::fundamentals_v2::__detail::__erase_nodes_if (_Container &__cont, _Predicate _↵
_pred)`

6.207.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly.

6.208 `erase_no_store_hash_fn_imps.hpp` File Reference

6.208.1 Detailed Description

Contains implementations of `cc_ht_map_`'s erase related functions, when the hash value is not stored.

6.209 `erase_no_store_hash_fn_imps.hpp` File Reference

6.209.1 Detailed Description

Contains implementations of `gp_ht_map_`'s erase related functions, when the hash value is not stored.

6.210 erase_store_hash_fn_imps.hpp File Reference

6.210.1 Detailed Description

Contains implementations of `cc_ht_map_'s` erase related functions, when the hash value is stored.

6.211 erase_store_hash_fn_imps.hpp File Reference

6.211.1 Detailed Description

Contains implementations of `gp_ht_map_'s` erase related functions, when the hash value is stored.

6.212 error_constants.h File Reference

Namespaces

- [std](#)

Enumerations

- `enum errc {`
 `address_family_not_supported, address_in_use, address_not_available, already_connected,`
 `argument_list_too_long, argument_out_of_domain, bad_address, bad_file_descriptor,`
 `broken_pipe, connection_aborted, connection_already_in_progress, connection_refused,`
 `connection_reset, cross_device_link, destination_address_required, device_or_resource_busy,`
 `directory_not_empty, executable_format_error, file_exists, file_too_large,`
 `filename_too_long, function_not_supported, host_unreachable, illegal_byte_sequence,`
 `inappropriate_io_control_operation, interrupted, invalid_argument, invalid_seek,`
 `io_error, is_a_directory, message_size, network_down,`
 `network_reset, network_unreachable, no_buffer_space, no_child_process,`
 `no_lock_available, no_message, no_protocol_option, no_space_on_device,`
 `no_such_device_or_address, no_such_device, no_such_file_or_directory, no_such_process,`
 `not_a_directory, not_a_socket, not_connected, not_enough_memory,`
 `operation_in_progress, operation_not_permitted, operation_not_supported, operation_would_block,`
 `permission_denied, protocol_not_supported, read_only_file_system, resource_deadlock_would_occur,`
 `resource_unavailable_try_again, result_out_of_range, timed_out, too_many_files_open_in_system,`
 `too_many_files_open, too_many_links, too_many_symbolic_link_levels, wrong_protocol_type }`
`}`

6.212.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<system_error>`.

6.213 exception File Reference

Classes

- class [std::bad_exception](#)
- class [std::exception](#)

Namespaces

- [__gnu_cxx](#)
- [std](#)

Macros

- `#define __cpp_lib_uncaught_exceptions`
- `#define __EXCEPTION__`

Typedefs

- typedef void(* [std::terminate_handler](#)) ()
- typedef void(* [std::unexpected_handler](#)) ()

Functions

- void [__gnu_cxx::__verbose_terminate_handler](#) ()
- terminate_handler [std::get_terminate](#) () noexcept
- unexpected_handler [std::get_unexpected](#) () noexcept
- terminate_handler [std::set_terminate](#) (terminate_handler) noexcept
- unexpected_handler [std::set_unexpected](#) (unexpected_handler) noexcept
- void [std::terminate](#) () noexcept `__attribute__((__noreturn__))`
- bool [std::uncaught_exception](#) () noexcept `__attribute__((__pure__))`
- int [std::uncaught_exceptions](#) () noexcept `__attribute__((__pure__))`
- void [std::unexpected](#) () `__attribute__((__noreturn__))`

6.213.1 Detailed Description

This is a Standard C++ Library header.

6.214 exception.hpp File Reference

Classes

- struct [__gnu_pbds::container_error](#)
- struct [__gnu_pbds::insert_error](#)
- struct [__gnu_pbds::join_error](#)
- struct [__gnu_pbds::resize_error](#)

Namespaces

- [__gnu_pbds](#)

Functions

- void [__gnu_pbds::__throw_container_error\(\)](#)
- void [__gnu_pbds::__throw_insert_error\(\)](#)
- void [__gnu_pbds::__throw_join_error\(\)](#)
- void [__gnu_pbds::__throw_resize_error\(\)](#)

6.214.1 Detailed Description

Contains exception classes.

6.215 exception_defines.h File Reference

Macros

- `#define __catch\(X\)`
- `#define __throw_exception_again`
- `#define __try`

6.215.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<exception>`.

6.216 exception_ptr.h File Reference

Classes

- class [std::__exception_ptr::exception_ptr](#)

Namespaces

- [std](#)

Functions

- `template<typename _Ex >`
`exception_ptr std::copy_exception (_Ex __ex) noexcept`
- `exception_ptr std::current_exception () noexcept`
- `template<typename _Ex >`
`exception_ptr std::make_exception_ptr (_Ex __ex) noexcept`
- `bool std::__exception_ptr::operator!= (const exception_ptr &, const exception_ptr &) noexcept __attribute__((__pure__))`
- `bool std::__exception_ptr::operator== (const exception_ptr &, const exception_ptr &) noexcept __attribute__((__pure__))`
- `void std::rethrow_exception (exception_ptr) __attribute__((__noreturn__))`
- `void std::__exception_ptr::swap (exception_ptr &__lhs, exception_ptr &__rhs)`

6.216.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<exception>`.

6.217 `extc++.h` File Reference

6.217.1 Detailed Description

This is an implementation file for a precompiled header.

6.218 `extptr_allocator.h` File Reference

Classes

- `class __gnu_cxx::ExtPtr_allocator<_Tp>`

Namespaces

- `__gnu_cxx`

Functions

- `template<typename _Tp >`
`void __gnu_cxx::swap (_ExtPtr_allocator<_Tp> &__larg, _ExtPtr_allocator<_Tp> &__rarg)`

6.218.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

Author

Bob Walters

An example allocator which uses an alternative pointer type from bits/pointer.h. Supports test cases which confirm container support for alternative pointers.

6.219 features.h File Reference

Macros

- [#define _GLIBCXX_BAL_QUICKSORT](#)
- [#define _GLIBCXX_FIND_CONSTANT_SIZE_BLOCKS](#)
- [#define _GLIBCXX_FIND_EQUAL_SPLIT](#)
- [#define _GLIBCXX_FIND_GROWING_BLOCKS](#)
- [#define _GLIBCXX_MERGESORT](#)
- [#define _GLIBCXX_QUICKSORT](#)
- [#define _GLIBCXX_TREE_DYNAMIC_BALANCING](#)
- [#define _GLIBCXX_TREE_FULL_COPY](#)
- [#define _GLIBCXX_TREE_INITIAL_SPLITTING](#)

6.219.1 Detailed Description

Defines on whether to include algorithm variants.

Less variants reduce executable size and compile time. This file is a GNU parallel extension to the Standard C++ Library.

6.219.2 Macro Definition Documentation

6.219.2.1 [#define _GLIBCXX_BAL_QUICKSORT](#)

Include parallel dynamically load-balanced quicksort.

See also

[__gnu_parallel::_Settings::sort_algorithm](#)

Definition at line 55 of file features.h.

6.219.2.2 `#define _GLIBCXX_FIND_CONSTANT_SIZE_BLOCKS`

Include the equal-sized blocks variant for `std::find`.

See also

`__gnu_parallel::_Settings::find_algorithm`

Definition at line 67 of file `features.h`.

6.219.2.3 `#define _GLIBCXX_FIND_EQUAL_SPLIT`

Include the equal splitting variant for `std::find`.

See also

`__gnu_parallel::_Settings::find_algorithm`

Definition at line 74 of file `features.h`.

6.219.2.4 `#define _GLIBCXX_FIND_GROWING_BLOCKS`

Include the growing blocks variant for `std::find`.

See also

`__gnu_parallel::_Settings::find_algorithm`

Definition at line 61 of file `features.h`.

6.219.2.5 `#define _GLIBCXX_MERGESORT`

Include parallel multi-way mergesort.

See also

`__gnu_parallel::_Settings::sort_algorithm`

Definition at line 41 of file `features.h`.

6.219.2.6 `#define _GLIBCXX_QUICKSORT`

Include parallel unbalanced quicksort.

See also

`__gnu_parallel::_Settings::sort_algorithm`

Definition at line 48 of file `features.h`.

6.219.2.7 `#define _GLIBCXX_TREE_DYNAMIC_BALANCING`

Include the dynamic balancing variant for `_Rb_tree::insert_unique(_Iter beg, _Iter __end)`.

See also

`__gnu_parallel::_Rb_tree`

Definition at line 91 of file `features.h`.

6.219.2.8 `#define _GLIBCXX_TREE_FULL_COPY`

In order to sort the input sequence of `_Rb_tree::insert_unique(_Iter beg, _Iter __end)` a full copy of the input elements is done.

See also

`__gnu_parallel::_Rb_tree`

Definition at line 100 of file `features.h`.

6.219.2.9 `#define _GLIBCXX_TREE_INITIAL_SPLITTING`

Include the initial splitting variant for `_Rb_tree::insert_unique(_Iter beg, _Iter __end)`.

See also

`__gnu_parallel::_Rb_tree`

Definition at line 83 of file `features.h`.

6.220 `fenv.h` File Reference

6.220.1 Detailed Description

This is a Standard C++ Library header.

6.221 `filesystem` File Reference

Namespaces

- [std](#)

Macros

- `#define __cpp_lib_experimental_filesystem`
- `#define _GLIBCXX_EXPERIMENTAL_FILESYSTEM`

6.221.1 Detailed Description

This is a TS C++ Library header.

6.222 find.h File Reference

Namespaces

- [__gnu_parallel](#)

Functions

- `template<typename _RAIter1, typename _RAIter2, typename _Pred, typename _Selector >`
`std::pair<_RAIter1, _RAIter2 > __gnu_parallel::__find_template (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred, _Selector __selector)`
- `template<typename _RAIter1, typename _RAIter2, typename _Pred, typename _Selector >`
`std::pair<_RAIter1, _RAIter2 > __gnu_parallel::__find_template (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred, _Selector __selector, equal_split_tag)`
- `template<typename _RAIter1, typename _RAIter2, typename _Pred, typename _Selector >`
`std::pair<_RAIter1, _RAIter2 > __gnu_parallel::__find_template (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred, _Selector __selector, growing_blocks_tag)`
- `template<typename _RAIter1, typename _RAIter2, typename _Pred, typename _Selector >`
`std::pair<_RAIter1, _RAIter2 > __gnu_parallel::__find_template (_RAIter1 __begin1, _RAIter1 __end1, _RAIter2 __begin2, _Pred __pred, _Selector __selector, constant_size_blocks_tag)`

6.222.1 Detailed Description

Parallel implementation base for `std::find()`, `std::equal()` and related functions. This file is a GNU parallel extension to the Standard C++ Library.

6.223 find_fn_imps.hpp File Reference

6.223.1 Detailed Description

Contains an implementation class for a binary_heap.

6.224 find_fn_imps.hpp File Reference

6.224.1 Detailed Description

Contains an implementation class for a base of binomial heaps.

6.225 find_fn_imps.hpp File Reference

6.225.1 Detailed Description

Contains an implementation class for bin_search_tree_.

6.226 find_fn_imps.hpp File Reference

6.226.1 Detailed Description

Contains implementations of cc_ht_map_'s find related functions.

6.227 find_fn_imps.hpp File Reference

6.227.1 Detailed Description

Contains implementations of gp_ht_map_'s find related functions.

6.228 find_fn_imps.hpp File Reference

6.228.1 Detailed Description

Contains implementations of lu_map_.

6.229 find_fn_imps.hpp File Reference

6.229.1 Detailed Description

Contains an implementation class for a pairing heap.

6.230 `find_fn_imps.hpp` File Reference

6.230.1 Detailed Description

Contains an implementation class for `pat_trie`.

6.231 `find_fn_imps.hpp` File Reference

6.231.1 Detailed Description

Contains an implementation for `rb_tree_`.

6.232 `find_fn_imps.hpp` File Reference

6.232.1 Detailed Description

Contains an implementation class for `splay_tree_`.

6.233 `find_fn_imps.hpp` File Reference

6.233.1 Detailed Description

Contains an implementation for `thin_heap_`.

6.234 `find_no_store_hash_fn_imps.hpp` File Reference

6.234.1 Detailed Description

Contains implementations of `gp_ht_map_`'s find related functions, when the hash value is not stored.

6.235 `find_selectors.h` File Reference

Classes

- struct [__gnu_parallel::__adjacent_find_selector](#)
- struct [__gnu_parallel::__find_first_of_selector<_FIterator>](#)
- struct [__gnu_parallel::__find_if_selector](#)
- struct [__gnu_parallel::__generic_find_selector](#)
- struct [__gnu_parallel::__mismatch_selector](#)

Namespaces

- [__gnu_parallel](#)

6.235.1 Detailed Description

_Function objects representing different tasks to be plugged into the parallel find algorithm. This file is a GNU parallel extension to the Standard C++ Library.

6.236 find_store_hash_fn_imps.hpp File Reference

6.236.1 Detailed Description

Contains implementations of cc_ht_map_'s find related functions, when the hash value is stored.

6.237 find_store_hash_fn_imps.hpp File Reference

6.237.1 Detailed Description

Contains implementations of gp_ht_map_'s insert related functions, when the hash value is stored.

6.238 for_each.h File Reference

Namespaces

- [__gnu_parallel](#)

Functions

- `template<typename _Iter, typename _UserOp, typename _Functionality, typename _Red, typename _Result >
_UserOp __gnu_parallel::__for_each_template_random_access (_Iter __begin, _Iter __end, _UserOp __user_op,
_Functionality &__functionality, _Red __reduction, _Result __reduction_start, _Result &__output, typename
std::iterator_traits< _Iter >::difference_type __bound, _Parallelism __parallelism_tag)`

6.238.1 Detailed Description

Main interface for embarrassingly parallel functions.

The explicit implementation are in other header files, like workstealing.h, par_loop.h, omp_loop.h, and omp_loop_↵static.h. This file is a GNU parallel extension to the Standard C++ Library.

6.239 `for_each_selectors.h` File Reference

Classes

- struct `__gnu_parallel::__accumulate_binop_reduct<_BinOp>`
- struct `__gnu_parallel::__accumulate_selector<_It>`
- struct `__gnu_parallel::__adjacent_difference_selector<_It>`
- struct `__gnu_parallel::__count_if_selector<_It, _Diff>`
- struct `__gnu_parallel::__count_selector<_It, _Diff>`
- struct `__gnu_parallel::__fill_selector<_It>`
- struct `__gnu_parallel::__for_each_selector<_It>`
- struct `__gnu_parallel::__generate_selector<_It>`
- struct `__gnu_parallel::__generic_for_each_selector<_It>`
- struct `__gnu_parallel::__identity_selector<_It>`
- struct `__gnu_parallel::__inner_product_selector<_It, _It2, _Tp>`
- struct `__gnu_parallel::__max_element_reduct<_Compare, _It>`
- struct `__gnu_parallel::__min_element_reduct<_Compare, _It>`
- struct `__gnu_parallel::__replace_if_selector<_It, _Op, _Tp>`
- struct `__gnu_parallel::__replace_selector<_It, _Tp>`
- struct `__gnu_parallel::__transform1_selector<_It>`
- struct `__gnu_parallel::__transform2_selector<_It>`
- struct `__gnu_parallel::__DummyReduct`
- struct `__gnu_parallel::__Nothing`

Namespaces

- `__gnu_parallel`

6.239.1 Detailed Description

Functors representing different tasks to be plugged into the generic parallelization methods for embarrassingly parallel functions. This file is a GNU parallel extension to the Standard C++ Library.

6.240 `formatter.h` File Reference

Classes

- class `__gnu_debug::__Safe_iterator<_Iterator, _Sequence>`
- class `__gnu_debug::__Safe_local_iterator<_Iterator, _Sequence>`
- class `__gnu_debug::__Safe_sequence<_Sequence>`

Namespaces

- `__gnu_debug`

Macros

- `#define _GLIBCXX_TYPEID(_Type)`

Enumerations

- `enum _Debug_msg_id {`
`__msg_valid_range, __msg_insert_singular, __msg_insert_different, __msg_erase_bad,`
`__msg_erase_different, __msg_subscript_oob, __msg_empty, __msg_unpartitioned,`
`__msg_unpartitioned_pred, __msg_unsorted, __msg_unsorted_pred, __msg_not_heap,`
`__msg_not_heap_pred, __msg_bad_bitset_write, __msg_bad_bitset_read, __msg_bad_bitset_flip,`
`__msg_self_splice, __msg_splice_alloc, __msg_splice_bad, __msg_splice_other,`
`__msg_splice_overlap, __msg_init_singular, __msg_init_copy_singular, __msg_init_const_singular,`
`__msg_copy_singular, __msg_bad_deref, __msg_bad_inc, __msg_bad_dec,`
`__msg_iter_subscript_oob, __msg_advance_oob, __msg_retreat_oob, __msg_iter_compare_bad,`
`__msg_compare_different, __msg_iter_order_bad, __msg_order_different, __msg_distance_bad,`
`__msg_distance_different, __msg_deref_istream, __msg_inc_istream, __msg_output_ostream,`
`__msg_deref_istreambuf, __msg_inc_istreambuf, __msg_insert_after_end, __msg_erase_after_bad,`
`__msg_valid_range2, __msg_local_iter_compare_bad, __msg_non_empty_range, __msg_self_move_↵`
`assign,`
`__msg_bucket_index_oob, __msg_valid_load_factor, __msg_equal_allocs, __msg_insert_range_from_↵`
`_self,`
`__msg_irreflexive_ordering }`

Functions

- `template<typename _Iterator >`
`bool __gnu_debug::__check_singular (const _Iterator &)`

6.240.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.241 forward_list File Reference

Macros

- `#define _GLIBCXX_FORWARD_LIST`

6.241.1 Detailed Description

This is a Standard C++ Library header.

6.242 forward_list File Reference

Classes

- class [__gnu_debug::Safe_forward_list<_SafeSequence>](#)
- class [std::__debug::forward_list<_Tp, _Alloc>](#)

Namespaces

- [__gnu_debug](#)
- [std](#)
- [std::__debug](#)

Macros

- `#define __glibcxx_check_valid_fl_range(_First, _Last, _Dist)`
- `#define _GLIBCXX_DEBUG_FORWARD_LIST`

Functions

- `template<typename _Tp, typename _Alloc>`
`bool std::__debug::operator!= (const forward_list<_Tp, _Alloc> &__lx, const forward_list<_Tp, _Alloc> &__ly)`
- `template<typename _Tp, typename _Alloc>`
`bool std::__debug::operator< (const forward_list<_Tp, _Alloc> &__lx, const forward_list<_Tp, _Alloc> &__ly)`
- `template<typename _Tp, typename _Alloc>`
`bool std::__debug::operator<= (const forward_list<_Tp, _Alloc> &__lx, const forward_list<_Tp, _Alloc> &__ly)`
- `template<typename _Tp, typename _Alloc>`
`bool std::__debug::operator== (const forward_list<_Tp, _Alloc> &__lx, const forward_list<_Tp, _Alloc> &__ly)`
- `template<typename _Tp, typename _Alloc>`
`bool std::__debug::operator> (const forward_list<_Tp, _Alloc> &__lx, const forward_list<_Tp, _Alloc> &__ly)`
- `template<typename _Tp, typename _Alloc>`
`bool std::__debug::operator>= (const forward_list<_Tp, _Alloc> &__lx, const forward_list<_Tp, _Alloc> &__ly)`
- `template<typename _Tp, typename _Alloc>`
`void std::__debug::swap (forward_list<_Tp, _Alloc> &__lx, forward_list<_Tp, _Alloc> &__ly) noexcept(noexcept(__lx.swap(__ly)))`

6.242.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.243 forward_list File Reference

Classes

- class [std::__profile::forward_list<_Tp, _Alloc>](#)

Namespaces

- [std](#)
- [std::__profile](#)

Macros

- `#define _GLIBCXX_PROFILE_FORWARD_LIST`

Functions

- `template<typename _Tp, typename _Alloc>`
`bool std::__profile::operator!= (const forward_list<_Tp, _Alloc> &__lx, const forward_list<_Tp, _Alloc> &__ly)`
- `template<typename _Tp, typename _Alloc>`
`bool std::__profile::operator< (const forward_list<_Tp, _Alloc> &__lx, const forward_list<_Tp, _Alloc> &__ly)`
- `template<typename _Tp, typename _Alloc>`
`bool std::__profile::operator<= (const forward_list<_Tp, _Alloc> &__lx, const forward_list<_Tp, _Alloc> &__ly)`
- `template<typename _Tp, typename _Alloc>`
`bool std::__profile::operator== (const forward_list<_Tp, _Alloc> &__lx, const forward_list<_Tp, _Alloc> &__ly)`
- `template<typename _Tp, typename _Alloc>`
`bool std::__profile::operator> (const forward_list<_Tp, _Alloc> &__lx, const forward_list<_Tp, _Alloc> &__ly)`
- `template<typename _Tp, typename _Alloc>`
`bool std::__profile::operator>= (const forward_list<_Tp, _Alloc> &__lx, const forward_list<_Tp, _Alloc> &__ly)`
- `template<typename _Tp, typename _Alloc>`
`void std::__profile::swap (forward_list<_Tp, _Alloc> &__lx, forward_list<_Tp, _Alloc> &__ly) noexcept(noexcept(__lx.swap(__ly)))`

6.243.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.244 forward_list File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_FORWARD_LIST`

Typedefs

- `template<typename _Tp >`
`using std::experimental::fundamentals_v2::pmr::forward_list = std::forward_list< _Tp, polymorphic_allocator< _Tp >>`

Functions

- `template<typename _Tp , typename _Alloc , typename _Up >`
`void std::experimental::fundamentals_v2::erase (forward_list< _Tp, _Alloc > &__cont, const _Up &__value)`
- `template<typename _Tp , typename _Alloc , typename _Predicate >`
`void std::experimental::fundamentals_v2::erase_if (forward_list< _Tp, _Alloc > &__cont, _Predicate __pred)`

6.244.1 Detailed Description

This is a TS C++ Library header.

6.245 [forward_list.h](#) File Reference

Classes

- struct [std::_Fwd_list_base](#)< _Tp, _Alloc >
- struct [std::_Fwd_list_const_iterator](#)< _Tp >
- struct [std::_Fwd_list_iterator](#)< _Tp >
- struct [std::_Fwd_list_node](#)< _Tp >
- struct [std::_Fwd_list_node_base](#)
- class [std::forward_list](#)< _Tp, _Alloc >

Namespaces

- [std](#)

Functions

- `template<typename _Tp >`
`bool std::operator!= (const _Fwd_list_iterator< _Tp > &__x, const _Fwd_list_const_iterator< _Tp > &__y) noexcept`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator!= (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator< (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator<= (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Tp >`
`bool std::operator== (const _Fwd_list_iterator< _Tp > &__x, const _Fwd_list_const_iterator< _Tp > &__y) noexcept`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator== (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator> (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator>= (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`
- `template<typename _Tp, typename _Alloc >`
`void std::swap (forward_list< _Tp, _Alloc > &__lx, forward_list< _Tp, _Alloc > &__ly) noexcept(noexcept(__lx.swap(__ly)))`

6.245.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<forward_list>`.

6.246 forward_list.tcc File Reference

Namespaces

- `std`

Macros

- `#define _FORWARD_LIST_TCC`

Functions

- `template<typename _Tp, typename _Alloc >`
`bool std::operator== (const forward_list< _Tp, _Alloc > &__lx, const forward_list< _Tp, _Alloc > &__ly)`

6.246.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<forward_list>`.

6.247 fs_dir.h File Reference

Namespaces

- [std](#)

Functions

- directory_iterator **std::experimental::filesystem::v1::begin** (directory_iterator __iter) noexcept
- directory_iterator **std::experimental::filesystem::v1::end** (directory_iterator) noexcept
- bool **std::experimental::filesystem::v1::operator==** (const directory_iterator &__lhs, const directory_iterator &__rhs)
- bool **std::experimental::filesystem::v1::operator!=** (const directory_iterator &__lhs, const directory_iterator &__rhs)
- recursive_directory_iterator **std::experimental::filesystem::v1::begin** (recursive_directory_iterator __iter) noexcept
- recursive_directory_iterator **std::experimental::filesystem::v1::end** (recursive_directory_iterator) noexcept
- bool **std::experimental::filesystem::v1::operator==** (const recursive_directory_iterator &__lhs, const recursive_directory_iterator &__rhs)
- bool **std::experimental::filesystem::v1::operator!=** (const recursive_directory_iterator &__lhs, const recursive_directory_iterator &__rhs)

6.247.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<experimental/filesystem>`.

6.248 fs_fwd.h File Reference

Namespaces

- [std](#)

Typedefs

- typedef chrono::time_point< chrono::system_clock > **std::experimental::filesystem::v1::file_time_type**

Enumerations

- enum `std::experimental::filesystem::v1::copy_options` : unsigned short { `none`, `skip_existing`, `overwrite_existing`, `update_existing`, `recursive`, `copy_symlinks`, `skip_symlinks`, `directories_only`, `create_symlinks`, `create_hard_links` }
- enum `directory_options` : unsigned char { `none`, `follow_directory_symlink`, `skip_permission_denied` }
- enum `file_type` : signed char { `none`, `not_found`, `regular`, `directory`, `symlink`, `block`, `character`, `fifo`, `socket`, `unknown` }
- enum `std::experimental::filesystem::v1::perms` : unsigned { `none`, `owner_read`, `owner_write`, `owner_exec`, `owner_all`, `group_read`, `group_write`, `group_exec`, `group_all`, `others_read`, `others_write`, `others_exec`, `others_all`, `all`, `set_uid`, `set_gid`, `sticky_bit`, `mask`, `unknown`, `add_perms`, `remove_perms`, `symlink_nofollow` }

Functions

- constexpr copy_options `std::experimental::filesystem::v1::operator&` (copy_options __x, copy_options __y) noexcept
- constexpr perms `std::experimental::filesystem::v1::operator&` (perms __x, perms __y) noexcept
- constexpr directory_options `std::experimental::filesystem::v1::operator&` (directory_options __x, directory_options __y) noexcept
- copy_options & `std::experimental::filesystem::v1::operator&=` (copy_options &__x, copy_options __y) noexcept
- perms & `std::experimental::filesystem::v1::operator&=` (perms &__x, perms __y) noexcept
- directory_options & `std::experimental::filesystem::v1::operator&=` (directory_options &__x, directory_options __y) noexcept
- constexpr copy_options `std::experimental::filesystem::v1::operator^` (copy_options __x, copy_options __y) noexcept
- constexpr perms `std::experimental::filesystem::v1::operator^` (perms __x, perms __y) noexcept
- constexpr directory_options `std::experimental::filesystem::v1::operator^` (directory_options __x, directory_options __y) noexcept
- copy_options & `std::experimental::filesystem::v1::operator^=` (copy_options &__x, copy_options __y) noexcept
- perms & `std::experimental::filesystem::v1::operator^=` (perms &__x, perms __y) noexcept
- directory_options & `std::experimental::filesystem::v1::operator^=` (directory_options &__x, directory_options __y) noexcept
- constexpr copy_options `std::experimental::filesystem::v1::operator|` (copy_options __x, copy_options __y) noexcept
- constexpr perms `std::experimental::filesystem::v1::operator|` (perms __x, perms __y) noexcept
- constexpr directory_options `std::experimental::filesystem::v1::operator|` (directory_options __x, directory_options __y) noexcept
- copy_options & `std::experimental::filesystem::v1::operator|=` (copy_options &__x, copy_options __y) noexcept
- perms & `std::experimental::filesystem::v1::operator|=` (perms &__x, perms __y) noexcept
- directory_options & `std::experimental::filesystem::v1::operator|=` (directory_options &__x, directory_options __y) noexcept

- constexpr copy_options **std::experimental::filesystem::v1::operator~** (copy_options __x) noexcept
- constexpr perms **std::experimental::filesystem::v1::operator~** (perms __x) noexcept
- constexpr directory_options **std::experimental::filesystem::v1::operator~** (directory_options __x) noexcept
- void **std::experimental::filesystem::v1::copy** (const path &__from, const path &__to, copy_options __options)
- void **std::experimental::filesystem::v1::copy** (const path &__from, const path &__to, copy_options __options, error_code &) noexcept
- bool **std::experimental::filesystem::v1::copy_file** (const path &__from, const path &__to, copy_options __↵ option)
- bool **std::experimental::filesystem::v1::copy_file** (const path &__from, const path &__to, copy_options __↵ option, error_code &) noexcept
- path **std::experimental::filesystem::v1::current_path** ()
- file_status **std::experimental::filesystem::v1::status** (const path &)
- file_status **std::experimental::filesystem::v1::status** (const path &, error_code &) noexcept
- bool **std::experimental::filesystem::v1::status_known** (file_status) noexcept
- file_status **std::experimental::filesystem::v1::symlink_status** (const path &)
- file_status **std::experimental::filesystem::v1::symlink_status** (const path &, error_code &) noexcept
- bool **std::experimental::filesystem::v1::is_regular_file** (file_status) noexcept
- bool **std::experimental::filesystem::v1::is_symlink** (file_status) noexcept

6.248.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<experimental/filesystem>`.

6.249 fs_path.h File Reference

Classes

- class [std::experimental::filesystem::v1::path](#)
- class [std::experimental::filesystem::v1::path::iterator](#)

Namespaces

- [std](#)

Typedefs

- template<typename _CharT, typename _Traits = std::char_traits<_CharT>>
using **std::experimental::filesystem::v1::__basic_string_view** = std::experimental::basic_string_view< _↵ CharT, _Traits >

Functions

- void `std::experimental::filesystem::v1::swap` (path &__lhs, path &__rhs) noexcept
- size_t `std::experimental::filesystem::v1::hash_value` (const path &__p) noexcept
- bool `std::experimental::filesystem::v1::operator<` (const path &__lhs, const path &__rhs) noexcept
- bool `std::experimental::filesystem::v1::operator<=` (const path &__lhs, const path &__rhs) noexcept
- bool `std::experimental::filesystem::v1::operator>` (const path &__lhs, const path &__rhs) noexcept
- bool `std::experimental::filesystem::v1::operator>=` (const path &__lhs, const path &__rhs) noexcept
- bool `std::experimental::filesystem::v1::operator==` (const path &__lhs, const path &__rhs) noexcept
- bool `std::experimental::filesystem::v1::operator!=` (const path &__lhs, const path &__rhs) noexcept
- path `std::experimental::filesystem::v1::operator/` (const path &__lhs, const path &__rhs)
- template<typename _CharT, typename _Traits >
basic_ostream< _CharT, _Traits > & `std::experimental::filesystem::v1::operator<<` (basic_ostream< _CharT, _Traits > &__os, const path &__p)
- template<typename _CharT, typename _Traits >
basic_istream< _CharT, _Traits > & `std::experimental::filesystem::v1::operator>>` (basic_istream< _CharT, _Traits > &__is, path &__p)
- template<typename _Source >
path `std::experimental::filesystem::v1::u8path` (const _Source &__source)
- template<typename _InputIterator >
path `std::experimental::filesystem::v1::u8path` (_InputIterator __first, _InputIterator __last)

6.249.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<experimental/filesystem>`.

6.249.2 Function Documentation

6.249.2.1 size_t std::experimental::filesystem::v1::hash_value (const path &__p) [noexcept]

Compare paths.

Referenced by `std::experimental::filesystem::v1::swap()`.

6.249.2.2 bool std::experimental::filesystem::v1::operator!= (const path &__lhs, const path &__rhs) [inline], [noexcept]

Compare paths.

Definition at line 507 of file `fs_path.h`.

6.249.2.3 path std::experimental::filesystem::v1::operator/ (const path &__lhs, const path &__rhs) [inline]

Append one path to another.

Definition at line 511 of file `fs_path.h`.

6.249.2.4 `bool std::experimental::filesystem::v1::operator< (const path & __lhs, const path & __rhs) [inline],
[noexcept]`

Compare paths.

Definition at line 487 of file fs_path.h.

6.249.2.5 `template<typename _CharT, typename _Traits > basic_ostream<_CharT, _Traits>& std::experimental::filesystem::v1::operator<< (basic_ostream<_CharT, _Traits> & __os, const path & __p)`

Write a path to a stream.

Definition at line 517 of file fs_path.h.

6.249.2.6 `bool std::experimental::filesystem::v1::operator<= (const path & __lhs, const path & __rhs) [inline],
[noexcept]`

Compare paths.

Definition at line 491 of file fs_path.h.

6.249.2.7 `bool std::experimental::filesystem::v1::operator== (const path & __lhs, const path & __rhs) [inline],
[noexcept]`

Compare paths.

Definition at line 503 of file fs_path.h.

6.249.2.8 `bool std::experimental::filesystem::v1::operator> (const path & __lhs, const path & __rhs) [inline],
[noexcept]`

Compare paths.

Definition at line 495 of file fs_path.h.

6.249.2.9 `bool std::experimental::filesystem::v1::operator>= (const path & __lhs, const path & __rhs) [inline],
[noexcept]`

Compare paths.

Definition at line 499 of file fs_path.h.

6.249.2.10 `template<typename _CharT, typename _Traits > basic_istream<_CharT, _Traits>& std::experimental::filesystem::v1::operator>> (basic_istream<_CharT, _Traits> & __is, path & __p)`

Read a path from a stream.

Definition at line 529 of file fs_path.h.

6.249.2.11 `void std::experimental::filesystem::v1::swap (path & __lhs, path & __rhs) [inline], [noexcept]`

Compare paths.

Definition at line 482 of file `fs_path.h`.

References `std::experimental::filesystem::v1::hash_value()`.

6.249.2.12 `template<typename _Source > path std::experimental::filesystem::v1::u8path (const _Source & __source) [inline]`

Compare paths.

Definition at line 542 of file `fs_path.h`.

References `std::experimental::filesystem::v1::u8path()`.

Referenced by `std::experimental::filesystem::v1::u8path()`.

6.249.2.13 `template<typename _InputIterator > path std::experimental::filesystem::v1::u8path (_InputIterator __first, _InputIterator __last) [inline]`

Compare paths.

Definition at line 554 of file `fs_path.h`.

References `std::basic_string< _CharT, _Traits, _Alloc >::data()`, `std::basic_string< _CharT, _Traits, _Alloc >::size()`, and `std::experimental::filesystem::v1::u8path()`.

6.250 **fstream File Reference**

Classes

- class [std::basic_filebuf< _CharT, _Traits >](#)
- class [std::basic_fstream< _CharT, _Traits >](#)
- class [std::basic_ifstream< _CharT, _Traits >](#)
- class [std::basic_ofstream< _CharT, _Traits >](#)

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_FSTREAM`

Functions

- `template<class _CharT, class _Traits >`
`void std::swap (basic_filebuf< _CharT, _Traits > &__x, basic_filebuf< _CharT, _Traits > &__y)`
- `template<class _CharT, class _Traits >`
`void std::swap (basic_ifstream< _CharT, _Traits > &__x, basic_ifstream< _CharT, _Traits > &__y)`
- `template<class _CharT, class _Traits >`
`void std::swap (basic_ofstream< _CharT, _Traits > &__x, basic_ofstream< _CharT, _Traits > &__y)`
- `template<class _CharT, class _Traits >`
`void std::swap (basic_fstream< _CharT, _Traits > &__x, basic_fstream< _CharT, _Traits > &__y)`

6.250.1 Detailed Description

This is a Standard C++ Library header.

6.251 fstream.tcc File Reference

Namespaces

- [std](#)

Macros

- `#define _FSTREAM_TCC`

6.251.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<fstream>`.

6.252 funtexcept.h File Reference

Namespaces

- [std](#)

Functions

- void **std::__throw_bad_alloc** (void) __attribute__((__noreturn__))
- void **std::__throw_bad_cast** (void) __attribute__((__noreturn__))
- void **std::__throw_bad_exception** (void) __attribute__((__noreturn__))
- void **std::__throw_bad_function_call** () __attribute__((__noreturn__))
- void **std::__throw_bad_typeid** (void) __attribute__((__noreturn__))
- void **std::__throw_domain_error** (const char *) __attribute__((__noreturn__))
- void **std::__throw_future_error** (int) __attribute__((__noreturn__))
- void **std::__throw_invalid_argument** (const char *) __attribute__((__noreturn__))
- void **std::__throw_ios_failure** (const char *) __attribute__((__noreturn__))
- void **std::__throw_length_error** (const char *) __attribute__((__noreturn__))
- void **std::__throw_logic_error** (const char *) __attribute__((__noreturn__))
- void **std::__throw_out_of_range** (const char *) __attribute__((__noreturn__))
- void **std::__throw_out_of_range_fmt** (const char *,...) __attribute__((__noreturn__)) __attribute__((__format__(__gnu_printf__)))
- void **std::__throw_overflow_error** (const char *) __attribute__((__noreturn__))
- void **std::__throw_range_error** (const char *) __attribute__((__noreturn__))
- void **std::__throw_runtime_error** (const char *) __attribute__((__noreturn__))
- void **std::__throw_system_error** (int) __attribute__((__noreturn__))
- void **std::__throw_underflow_error** (const char *) __attribute__((__noreturn__))

6.252.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<exception>`.

This header provides support for `-fno-exceptions`.

6.253 functional File Reference

Classes

- struct **std::__is_location_invariant**< _Tp >
- struct **std::_Bind**< _Signature >
- struct **std::_Bind_result**< _Result, _Signature >
- class **std::_Function_base**
- struct **std::_Maybe_get_result_type**< _Functor, typename >
- struct **std::_Maybe_unary_or_binary_function**< _Res, _ArgTypes >
- struct **std::_Maybe_unary_or_binary_function**< _Res, _T1 >
- struct **std::_Maybe_unary_or_binary_function**< _Res, _T1, _T2 >
- struct **std::_Maybe_wrap_member_pointer**< _Tp >
- struct **std::_Maybe_wrap_member_pointer**< _Tp _Class::* >
- class **std::_Mu**< _Arg, _IsBindExp, _IsPlaceholder >
- class **std::_Mu**< _Arg, false, false >
- class **std::_Mu**< _Arg, false, true >
- class **std::_Mu**< _Arg, true, false >
- class **std::_Mu**< reference_wrapper< _Tp >, false, false >

- struct `std::_Placeholder<_Num>`
- struct `std::_Reference_wrapper_base<_Tp>`
- struct `std::_Reference_wrapper_base_impl<_Unary, _Binary, _Tp>`
- struct `std::_Weak_result_type<_Functor>`
- struct `std::_Weak_result_type_impl<_Functor>`
- struct `std::_Weak_result_type_impl<_Res(&)(_ArgTypes...)>`
- struct `std::_Weak_result_type_impl<_Res(*)(_ArgTypes...)>`
- struct `std::_Weak_result_type_impl<_Res(_ArgTypes...)>`
- struct `std::_Weak_result_type_impl<_Res(_Class::*)(_ArgTypes...) const>`
- struct `std::_Weak_result_type_impl<_Res(_Class::*)(_ArgTypes...) const volatile>`
- struct `std::_Weak_result_type_impl<_Res(_Class::*)(_ArgTypes...) volatile>`
- struct `std::_Weak_result_type_impl<_Res(_Class::*)(_ArgTypes...)>`
- class `std::bad_function_call`
- class `std::function<_Res(_ArgTypes...)>`
- struct `std::is_bind_expression<_Tp>`
- struct `std::is_bind_expression<_Bind<_Signature>>`
- struct `std::is_bind_expression<_Bind_result<_Result, _Signature>>`
- struct `std::is_bind_expression<const _Bind<_Signature>>`
- struct `std::is_bind_expression<const _Bind_result<_Result, _Signature>>`
- struct `std::is_bind_expression<const volatile _Bind<_Signature>>`
- struct `std::is_bind_expression<const volatile _Bind_result<_Result, _Signature>>`
- struct `std::is_bind_expression<volatile _Bind<_Signature>>`
- struct `std::is_bind_expression<volatile _Bind_result<_Result, _Signature>>`
- struct `std::is_placeholder<_Tp>`
- struct `std::is_placeholder<_Placeholder<_Num>>`
- class `std::reference_wrapper<_Tp>`

Namespaces

- `std`
- `std::placeholders`

Macros

- `#define _GLIBCXX_FUNCTIONAL`
- `#define _GLIBCXX_MEM_FN_TRAITS(_REF, _LVAL, _RVAL)`
- `#define _GLIBCXX_MEM_FN_TRAITS2(_CV, _REF, _LVAL, _RVAL)`

Typedefs

- `template<typename _From, typename _To>`
`using std::__check_func_return_type = __or_< is_void<_To>, is_same<_From, _To>, is_convertible<_From, _To>>`
- `template<typename _Tp, typename _Tp2 = typename decay<_Tp>::type>`
`using std::__is_socketlike = __or_< is_integral<_Tp2>, is_enum<_Tp2>>`
- `template<typename _Tp1, typename _Tp2>`
`using std::NotSame = __not_< is_same<typename std::decay<_Tp1>::type, typename std::decay<_Tp2>::type>>`
- `template<std::size_t __i, typename _Tuple>`
`using std::Safe_tuple_element_t = typename enable_if<(__i< tuple_size<_Tuple>::value), tuple_element<__i, _Tuple>>::type::type`

Enumerations

- enum **_Manager_operation** { **__get_type_info**, **__get_func_ptr**, **__clone_func_ptr**, **__destroy_func_ptr** }

Functions

- template<typename _Callable, typename... _Args>
_Bind_simple_helper< _Callable, _Args... >::__type **std::__bind_simple** (_Callable &&__callable, _Args &&...__args)
- template<typename _Functor >
_Functor & **std::__callable_func_ptr** (_Functor &__f)
- template<typename _Member, typename _Class >
_Mem_fn< _Member _Class::* > **std::__callable_func_ptr** (_Member _Class::*__p)
- template<typename _Member, typename _Class >
_Mem_fn< _Member _Class::* > **std::__callable_func_ptr** (_Member _Class::*const &__p)
- template<typename _Member, typename _Class >
_Mem_fn< _Member _Class::* > **std::__callable_func_ptr** (_Member _Class::*volatile &__p)
- template<typename _Member, typename _Class >
_Mem_fn< _Member _Class::* > **std::__callable_func_ptr** (_Member _Class::*const volatile &__p)
- template<typename _Tp, typename _Up = typename __inv_unwrap<_Tp>::type>
_Up && **std::__invfwd** (typename remove_reference< _Tp >::type &__t) noexcept
- template<typename _Callable, typename... _Args>
result_of< _Callable &&(_Args &&...)>::type **std::__invoke** (_Callable &&__fn, _Args &&...__args)
- template<typename _Res, typename _Fn, typename... _Args>
_Res **std::__invoke_impl** (__invoke_other, _Fn &&__f, _Args &&...__args) noexcept(noexcept(**std::forward**< __
_Fn >(__f)(**std::forward**< _Args >(__args)...))
- template<typename _Res, typename _MemFun, typename _Tp, typename... _Args>
_Res **std::__invoke_impl** (__invoke_memfun_ref, _MemFun &&__f, _Tp &&__t, _Args &&...__args)
noexcept(noexcept((__invfwd< _Tp >(__t).*_f)(**std::forward**< _Args >(__args)...)))
- template<typename _Res, typename _MemFun, typename _Tp, typename... _Args>
_Res **std::__invoke_impl** (__invoke_memfun_deref, _MemFun &&__f, _Tp &&__t, _Args &&...__args)
noexcept(noexcept(((**std::forward**< _Tp >(__t).*_f)(**std::forward**< _Args >(__args)...)))
- template<typename _Res, typename _MemPtr, typename _Tp >
_Res **std::__invoke_impl** (__invoke_memobj_ref, _MemPtr &&__f, _Tp &&__t) noexcept(noexcept(__invfwd<
_Tp >(__t).*_f))
- template<typename _Res, typename _MemPtr, typename _Tp >
_Res **std::__invoke_impl** (__invoke_memobj_deref, _MemPtr &&__f, _Tp &&__t) noexcept(noexcept((**std::for-**
ward< _Tp >(__t).*_f))
- template<std::size_t _Ind, typename... _Tp>
auto **std::__volget** (volatile tuple< _Tp... > &__tuple) -> __tuple_element_t< _Ind, tuple< _Tp... >> volatile &
- template<std::size_t _Ind, typename... _Tp>
auto **std::__volget** (const volatile tuple< _Tp... > &__tuple) -> __tuple_element_t< _Ind, tuple< _Tp... >>
const volatile &
- template<typename _Func, typename... _BoundArgs>
_Bind_helper< __is_socketlike< _Func >::value, _Func, _BoundArgs... >::type **std::bind** (_Func &&__f, __
BoundArgs &&...__args)
- template<typename _Result, typename _Func, typename... _BoundArgs>
_Bindres_helper< _Result, _Func, _BoundArgs... >::type **std::bind** (_Func &&__f, _BoundArgs &&...__args)
- template<typename _Tp, typename _Class >
_Mem_fn< _Tp _Class::* > **std::mem_fn** (_Tp _Class::*__pm) noexcept

- `template<typename _Res, typename... _Args>`
`bool std::operator!= (const function< _Res(_Args...)> &__f, nullptr_t) noexcept`
- `template<typename _Res, typename... _Args>`
`bool std::operator!= (nullptr_t, const function< _Res(_Args...)> &__f) noexcept`
- `template<typename _Res, typename... _Args>`
`bool std::operator== (const function< _Res(_Args...)> &__f, nullptr_t) noexcept`
- `template<typename _Res, typename... _Args>`
`bool std::operator== (nullptr_t, const function< _Res(_Args...)> &__f) noexcept`
- `template<typename _Res, typename... _Args>`
`void std::swap (function< _Res(_Args...)> &__x, function< _Res(_Args...)> &__y) noexcept`

- `template<typename _Tp >`
`reference_wrapper< _Tp > std::ref (_Tp &__t) noexcept`
- `template<typename _Tp >`
`reference_wrapper< const _Tp > std::cref (const _Tp &__t) noexcept`
- `template<typename _Tp >`
`void std::ref (const _Tp &&)=delete`
- `template<typename _Tp >`
`void std::cref (const _Tp &&)=delete`
- `template<typename _Tp >`
`reference_wrapper< _Tp > std::ref (reference_wrapper< _Tp > __t) noexcept`
- `template<typename _Tp >`
`reference_wrapper< const _Tp > std::cref (reference_wrapper< _Tp > __t) noexcept`

Variables

- `const _Placeholder< 1 > std::placeholders::_1`
- `const _Placeholder< 10 > std::placeholders::_10`
- `const _Placeholder< 11 > std::placeholders::_11`
- `const _Placeholder< 12 > std::placeholders::_12`
- `const _Placeholder< 13 > std::placeholders::_13`
- `const _Placeholder< 14 > std::placeholders::_14`
- `const _Placeholder< 15 > std::placeholders::_15`
- `const _Placeholder< 16 > std::placeholders::_16`
- `const _Placeholder< 17 > std::placeholders::_17`
- `const _Placeholder< 18 > std::placeholders::_18`
- `const _Placeholder< 19 > std::placeholders::_19`
- `const _Placeholder< 2 > std::placeholders::_2`
- `const _Placeholder< 20 > std::placeholders::_20`
- `const _Placeholder< 21 > std::placeholders::_21`
- `const _Placeholder< 22 > std::placeholders::_22`
- `const _Placeholder< 23 > std::placeholders::_23`
- `const _Placeholder< 24 > std::placeholders::_24`
- `const _Placeholder< 25 > std::placeholders::_25`
- `const _Placeholder< 26 > std::placeholders::_26`
- `const _Placeholder< 27 > std::placeholders::_27`
- `const _Placeholder< 28 > std::placeholders::_28`
- `const _Placeholder< 29 > std::placeholders::_29`
- `const _Placeholder< 3 > std::placeholders::_3`
- `const _Placeholder< 4 > std::placeholders::_4`

- `const _Placeholder< 5 > std::placeholders::_5`
- `const _Placeholder< 6 > std::placeholders::_6`
- `const _Placeholder< 7 > std::placeholders::_7`
- `const _Placeholder< 8 > std::placeholders::_8`
- `const _Placeholder< 9 > std::placeholders::_9`

6.253.1 Detailed Description

This is a Standard C++ Library header.

6.254 functional File Reference

Classes

- class [__gnu_cxx::binary_compose< _Operation1, _Operation2, _Operation3 >](#)
- struct [__gnu_cxx::constant_binary_fun< _Result, _Arg1, _Arg2 >](#)
- struct [__gnu_cxx::constant_unary_fun< _Result, _Argument >](#)
- struct [__gnu_cxx::constant_void_fun< _Result >](#)
- struct [__gnu_cxx::project1st< _Arg1, _Arg2 >](#)
- struct [__gnu_cxx::project2nd< _Arg1, _Arg2 >](#)
- struct [__gnu_cxx::select1st< _Pair >](#)
- struct [__gnu_cxx::select2nd< _Pair >](#)
- class [__gnu_cxx::subtractive_rng](#)
- class [__gnu_cxx::unary_compose< _Operation1, _Operation2 >](#)

Namespaces

- [__gnu_cxx](#)

Macros

- `#define _EXT_FUNCTIONAL`

Functions

- `template<class _Operation1 , class _Operation2 >`
`unary_compose< _Operation1, _Operation2 > __gnu_cxx::compose1 (const _Operation1 &__fn1, const _↵`
`Operation2 &__fn2)`
- `template<class _Operation1 , class _Operation2 , class _Operation3 >`
`binary_compose< _Operation1, _Operation2, _Operation3 > __gnu_cxx::compose2 (const _Operation1 &__fn1,`
`const _Operation2 &__fn2, const _Operation3 &__fn3)`
- `template<class _Result >`
`constant_void_fun< _Result > __gnu_cxx::constant0 (const _Result &__val)`
- `template<class _Result >`
`constant_unary_fun< _Result, _Result > __gnu_cxx::constant1 (const _Result &__val)`

- `template<class _Result >`
`constant_binary_fun< _Result, _Result, _Result > __gnu_cxx::constant2 (const _Result &__val)`
- `template<class _Tp >`
`_Tp __gnu_cxx::identity_element (std::plus< _Tp >)`
- `template<class _Tp >`
`_Tp __gnu_cxx::identity_element (std::multiplies< _Tp >)`
- `template<class _Ret, class _Tp, class _Arg >`
`mem_fun1_t< _Ret, _Tp, _Arg > __gnu_cxx::mem_fun1 (_Ret(_Tp::*__f)(_Arg))`
- `template<class _Ret, class _Tp, class _Arg >`
`const_mem_fun1_t< _Ret, _Tp, _Arg > __gnu_cxx::mem_fun1 (_Ret(_Tp::*__f)(_Arg) const)`
- `template<class _Ret, class _Tp, class _Arg >`
`mem_fun1_ref_t< _Ret, _Tp, _Arg > __gnu_cxx::mem_fun1_ref (_Ret(_Tp::*__f)(_Arg))`
- `template<class _Ret, class _Tp, class _Arg >`
`const_mem_fun1_ref_t< _Ret, _Tp, _Arg > __gnu_cxx::mem_fun1_ref (_Ret(_Tp::*__f)(_Arg) const)`

6.254.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGL STL subset).

6.255 functional File Reference

Classes

- class `std::experimental::fundamentals_v2::_Not_fn< _Fn >`

Namespaces

- `std`

Macros

- `#define __cpp_lib_experimental_boyer_moore_searching`
- `#define __cpp_lib_experimental_not_fn`
- `#define _GLIBCXX_EXPERIMENTAL_FUNCTIONAL`

Typedefs

- `template<typename _RAIter, typename _Hash, typename _Pred, typename _Val = typename iterator_traits<_RAIter>::value_type, typename _Diff = typename iterator_traits<_RAIter>::difference_type>`
`using std::experimental::fundamentals_v1::_boyer_moore_base_t = std::conditional_t< sizeof(_Val)==1`
`&&is_integral< _Val >::value &&__is_std_equal_to< _Pred >::value, __boyer_moore_array_base< _Diff, 256,`
`_Pred >, __boyer_moore_map_base< _Val, _Diff, _Hash, _Pred >>`

Functions

- `template<typename _RAIter, typename _Hash = std::hash<typename std::iterator_traits<_RAIter>::value_type>, typename _BinaryPredicate = equal_to<>>>`
`boyer_moore_horspool_searcher< _RAIter, _Hash, _BinaryPredicate > std::experimental::fundamentals_v1::make_boyer_moore_horspool_searcher (_RAIter __pat_first, _RAIter __pat_last, _Hash __hf=_Hash(), _BinaryPredicate __pred=_BinaryPredicate())`
- `template<typename _RAIter, typename _Hash = std::hash<typename std::iterator_traits<_RAIter>::value_type>, typename _BinaryPredicate = equal_to<>>>`
`boyer_moore_searcher< _RAIter, _Hash, _BinaryPredicate > std::experimental::fundamentals_v1::make_boyer_moore_searcher (_RAIter __pat_first, _RAIter __pat_last, _Hash __hf=_Hash(), _BinaryPredicate __pred=_BinaryPredicate())`
- `template<typename _ForwardIterator, typename _BinaryPredicate = std::equal_to<>>>`
`default_searcher< _ForwardIterator, _BinaryPredicate > std::experimental::fundamentals_v1::make_default_searcher (_ForwardIterator __pat_first, _ForwardIterator __pat_last, _BinaryPredicate __pred=_BinaryPredicate())`
- `template<typename _Fn >`
`auto std::experimental::fundamentals_v2::not_fn (_Fn &&__fn) noexcept(std::is_nothrow_constructible< std::decay_t< _Fn >, _Fn && >::value)`

Variables

- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_bind_expression_v`
- `template<typename _Tp >`
`constexpr int std::experimental::fundamentals_v1::is_placeholder_v`

6.255.1 Detailed Description

This is a TS C++ Library header.

6.255.2 Function Documentation

- 6.255.2.1 `template<typename _RAIter, typename _Hash = std::hash<typename std::iterator_traits<_RAIter>::value_type>, typename _BinaryPredicate = equal_to<>>> boyer_moore_horspool_searcher<_RAIter, _Hash, _BinaryPredicate>`
`std::experimental::fundamentals_v1::make_boyer_moore_horspool_searcher (_RAIter __pat_first, _RAIter __pat_last, _Hash __hf=_Hash(), _BinaryPredicate __pred=_BinaryPredicate()) [inline]`

Generator function for boyer_moore_horspool_searcher.

Definition at line 311 of file experimental/functional.

- 6.255.2.2 `template<typename _RAIter, typename _Hash = std::hash<typename std::iterator_traits<_RAIter>::value_type>, typename _BinaryPredicate = equal_to<>>> boyer_moore_searcher<_RAIter, _Hash, _BinaryPredicate>`
`std::experimental::fundamentals_v1::make_boyer_moore_searcher (_RAIter __pat_first, _RAIter __pat_last, _Hash __hf=_Hash(), _BinaryPredicate __pred=_BinaryPredicate()) [inline]`

Generator function for boyer_moore_searcher.

Definition at line 301 of file experimental/functional.

6.255.2.3 `template<typename _ForwardIterator, typename _BinaryPredicate = std::equal_to<>> default_searcher<_ForwardIterator, _BinaryPredicate> std::experimental::fundamentals_v1::make_default_searcher (_ForwardIterator __pat_first, _ForwardIterator __pat_last, _BinaryPredicate __pred = _BinaryPredicate()) [inline]`

Generator function for `default_searcher`.

Definition at line 291 of file `experimental/functional`.

6.255.2.4 `template<typename _Fn > auto std::experimental::fundamentals_v2::not_fn (_Fn && __fn) [inline], [noexcept]`

[func.not_fn] Function template `not_fn`

Definition at line 430 of file `experimental/functional`.

6.255.3 Variable Documentation

6.255.3.1 `template<typename _Tp > constexpr bool std::experimental::fundamentals_v1::is_bind_expression_v`

Variable template for `std::is_bind_expression`.

Definition at line 62 of file `experimental/functional`.

6.255.3.2 `template<typename _Tp > constexpr int std::experimental::fundamentals_v1::is_placeholder_v`

Variable template for `std::is_placeholder`.

Definition at line 66 of file `experimental/functional`.

6.256 functional_hash.h File Reference

Classes

- struct `std::hash< _Tp >`
- struct `std::hash< _Tp >`
- struct `std::hash< _Tp * >`
- struct `std::hash< bool >`
- struct `std::hash< char >`
- struct `std::hash< char16_t >`
- struct `std::hash< char32_t >`
- struct `std::hash< double >`
- struct `std::hash< float >`
- struct `std::hash< int >`
- struct `std::hash< long >`
- struct `std::hash< long double >`
- struct `std::hash< long long >`
- struct `std::hash< short >`
- struct `std::hash< signed char >`
- struct `std::hash< unsigned char >`
- struct `std::hash< unsigned int >`
- struct `std::hash< unsigned long >`
- struct `std::hash< unsigned long long >`
- struct `std::hash< unsigned short >`
- struct `std::hash< wchar_t >`

Namespaces

- [std](#)

Macros

- `#define _Cxx_hashtable_define_trivial_hash(_Tp)`

6.256.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<functional>`.

6.257 functions.h File Reference

Classes

- class [__gnu_debug::Safe_iterator<_Iterator, _Sequence>](#)

Namespaces

- [__gnu_debug](#)

Functions

- `template<typename _Iterator >`
`bool __gnu_debug::__check_dereferenceable (const _Iterator &)`
- `template<typename _Tp >`
`bool __gnu_debug::__check_dereferenceable (const _Tp *__ptr)`
- `template<typename _ForwardIterator, typename _Tp >`
`bool __gnu_debug::__check_partitioned_lower (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value)`
- `template<typename _ForwardIterator, typename _Tp, typename _Pred >`
`bool __gnu_debug::__check_partitioned_lower (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value, _Pred __pred)`
- `template<typename _ForwardIterator, typename _Tp >`
`bool __gnu_debug::__check_partitioned_upper (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value)`
- `template<typename _ForwardIterator, typename _Tp, typename _Pred >`
`bool __gnu_debug::__check_partitioned_upper (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value, _Pred __pred)`
- `template<typename _Iterator >`
`bool __gnu_debug::__check_singular (const _Iterator &)`
- `template<typename _Tp >`
`bool __gnu_debug::__check_singular (const _Tp *__ptr)`
- `bool __gnu_debug::__check_singular_aux (const void *)`

- `template<typename _InputIterator >`
`bool __gnu_debug::__check_sorted (const _InputIterator &__first, const _InputIterator &__last)`
- `template<typename _InputIterator, typename _Predicate >`
`bool __gnu_debug::__check_sorted (const _InputIterator &__first, const _InputIterator &__last, _Predicate __pred)`
- `template<typename _InputIterator >`
`bool __gnu_debug::__check_sorted_aux (const _InputIterator &, const _InputIterator &, std::input_iterator_tag)`
- `template<typename _ForwardIterator >`
`bool __gnu_debug::__check_sorted_aux (_ForwardIterator __first, _ForwardIterator __last, std::forward_iterator_tag)`
- `template<typename _InputIterator, typename _Predicate >`
`bool __gnu_debug::__check_sorted_aux (const _InputIterator &, const _InputIterator &, _Predicate, std::input_iterator_tag)`
- `template<typename _ForwardIterator, typename _Predicate >`
`bool __gnu_debug::__check_sorted_aux (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred, std::forward_iterator_tag)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`bool __gnu_debug::__check_sorted_set (const _InputIterator1 &__first, const _InputIterator1 &__last, const _InputIterator2 &)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Predicate >`
`bool __gnu_debug::__check_sorted_set (const _InputIterator1 &__first, const _InputIterator1 &__last, const _InputIterator2 &, _Predicate __pred)`
- `template<typename _InputIterator >`
`bool __gnu_debug::__check_sorted_set_aux (const _InputIterator &__first, const _InputIterator &__last, std::__true_type)`
- `template<typename _InputIterator >`
`bool __gnu_debug::__check_sorted_set_aux (const _InputIterator &, const _InputIterator &, std::__false_type)`
- `template<typename _InputIterator, typename _Predicate >`
`bool __gnu_debug::__check_sorted_set_aux (const _InputIterator &__first, const _InputIterator &__last, _Predicate __pred, std::__true_type)`
- `template<typename _InputIterator, typename _Predicate >`
`bool __gnu_debug::__check_sorted_set_aux (const _InputIterator &, const _InputIterator &, _Predicate, std::__false_type)`
- `template<typename _CharT, typename _Integer >`
`const _CharT * __gnu_debug::__check_string (const _CharT *__s, const _Integer &__n __attribute__\(\(__unused__\)\))`
- `template<typename _CharT >`
`const _CharT * __gnu_debug::__check_string (const _CharT *__s)`
- `template<typename _InputIterator >`
`_InputIterator __gnu_debug::__check_valid_range (const _InputIterator &__first, const _InputIterator &__last __attribute__\(\(__unused__\)\))`
- `template<typename _Iterator, typename _Sequence, typename _InputIterator >`
`bool __gnu_debug::__foreign_iterator (const _Safe_iterator< _Iterator, _Sequence > &__it, _InputIterator __other, _InputIterator __other_end)`
- `template<typename _Iterator, typename _Sequence, typename _Integral >`
`bool __gnu_debug::__foreign_iterator_aux (const _Safe_iterator< _Iterator, _Sequence > &, _Integral, std::__true_type)`
- `template<typename _Iterator, typename _Sequence, typename _InputIterator >`
`bool __gnu_debug::__foreign_iterator_aux (const _Safe_iterator< _Iterator, _Sequence > &__it, _InputIterator __other, _InputIterator __other_end, std::__false_type)`
- `template<typename _Iterator, typename _Sequence, typename _OtherIterator >`
`bool __gnu_debug::__foreign_iterator_aux2 (const _Safe_iterator< _Iterator, _Sequence > &__it, const _Safe_iterator< _OtherIterator, _Sequence > &__other, const _Safe_iterator< _OtherIterator, _Sequence > &)`

- `template<typename _Iterator, typename _Sequence, typename _OtherIterator, typename _OtherSequence >`
`bool __gnu_debug::__foreign_iterator_aux2 (const _Safe_iterator< _Iterator, _Sequence > &__it, const _Safe_↵`
`_iterator< _OtherIterator, _OtherSequence > &, const _Safe_iterator< _OtherIterator, _OtherSequence > &)`
- `template<typename _Iterator, typename _Sequence, typename _InputIterator >`
`bool __gnu_debug::__foreign_iterator_aux2 (const _Safe_iterator< _Iterator, _Sequence > &__it, const _↵`
`InputIterator &__other, const _InputIterator &__other_end)`
- `template<typename _Iterator, typename _Sequence, typename _InputIterator >`
`bool __gnu_debug::__foreign_iterator_aux3 (const _Safe_iterator< _Iterator, _Sequence > &__it, const _↵`
`InputIterator &__other, const _InputIterator &__other_end, std::__true_type)`
- `template<typename _Iterator, typename _Sequence, typename _InputIterator >`
`bool __gnu_debug::__foreign_iterator_aux3 (const _Safe_iterator< _Iterator, _Sequence > &, const _Input_↵`
`Iterator &, const _InputIterator &, std::__false_type)`
- `template<typename _Iterator, typename _Sequence >`
`bool __gnu_debug::__foreign_iterator_aux4 (const _Safe_iterator< _Iterator, _Sequence > &__it, const type-↵`
`name _Sequence::value_type *__other)`
- `template<typename _Iterator, typename _Sequence >`
`bool __gnu_debug::__foreign_iterator_aux4 (const _Safe_iterator< _Iterator, _Sequence > &,...)`
- `template<typename _Iterator >`
`bool __gnu_debug::__is_irreflexive (_Iterator __it)`
- `template<typename _Iterator, typename _Pred >`
`bool __gnu_debug::__is_irreflexive_pred (_Iterator __it, _Pred __pred)`

6.257.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.258 future File Reference

Classes

- class `std::__basic_future< _Res >`
- struct `std::__future_base`
- struct `std::__future_base::Result< _Res >`
- struct `std::__future_base::Result< _Res & >`
- struct `std::__future_base::Result< void >`
- struct `std::__future_base::Result_alloc< _Res, _Alloc >`
- struct `std::__future_base::Result_base`
- class `std::future< _Res >`
- class `std::future< _Res >`
- class `std::future< _Res & >`
- class `std::future< void >`
- class `std::future_error`
- struct `std::is_error_code_enum< future_errc >`
- class `std::packaged_task< _Res(_ArgTypes...)>`
- class `std::promise< _Res >`
- class `std::promise< _Res >`
- class `std::promise< _Res & >`
- class `std::promise< void >`
- class `std::shared_future< _Res >`
- class `std::shared_future< _Res >`
- class `std::shared_future< _Res & >`
- class `std::shared_future< void >`

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_FUTURE`

Typedefs

- `template<typename _Fn, typename... _Args>`
`using std::__async_result_of = typename result_of< typename decay< _Fn >::type(typename decay< _Args >::type...)>::type`

Enumerations

- `enum std::future_errc { future_already_retrieved, promise_already_satisfied, no_state, broken_promise }`
- `enum std::future_status { ready, timeout, deferred }`
- `enum std::launch { async, deferred }`

Functions

- `template<typename _Signature, typename _Fn, typename _Alloc >`
`static shared_ptr< __future_base::__Task_state_base< _Signature > > std::__create_task_state (_Fn &&__fn, const _Alloc &__a)`
- `template<typename _Fn, typename... _Args>`
`future< __async_result_of< _Fn, _Args... > > std::async (launch __policy, _Fn &&__fn, _Args &&...__args)`
- `template<typename _Fn, typename... _Args>`
`future< __async_result_of< _Fn, _Args... > > std::async (_Fn &&__fn, _Args &&...__args)`
- `const error_category & std::future_category () noexcept`
- `error_code std::make_error_code (future_errc __errc) noexcept`
- `error_condition std::make_error_condition (future_errc __errc) noexcept`
- `constexpr launch std::operator& (launch __x, launch __y)`
- `launch & std::operator&= (launch &__x, launch __y)`
- `constexpr launch std::operator^ (launch __x, launch __y)`
- `launch & std::operator^= (launch &__x, launch __y)`
- `constexpr launch std::operator| (launch __x, launch __y)`
- `launch & std::operator|= (launch &__x, launch __y)`
- `constexpr launch std::operator~ (launch __x)`
- `template<typename _Res >`
`void std::swap (promise< _Res > &__x, promise< _Res > &__y) noexcept`
- `template<typename _Res, typename... _ArgTypes>`
`void std::swap (packaged_task< _Res(_ArgTypes...)> &__x, packaged_task< _Res(_ArgTypes...)> &__y) noexcept`

6.258.1 Detailed Description

This is a Standard C++ Library header.

6.259 gp_ht_map_.hpp File Reference

Classes

- class [__gnu_pbds::detail::gp_ht_map](#)< Key, Mapped, Hash_Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe_Fn, Probe_Fn, Resize_Policy >

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_GEN_POS`
- `#define PB_DS_GP_HASH_NAME`
- `#define PB_DS_GP_HASH_TRAITS_BASE`
- `#define PB_DS_HASH_EQ_FN_C_DEC`
- `#define PB_DS_RANGED_PROBE_FN_C_DEC`

Variables

- `empty_entry_status`
- `erased_entry_status`
- `valid_entry_status`

6.259.1 Detailed Description

Contains an implementation class for general probing hash.

6.260 gsllice.h File Reference

Classes

- class [std::gsllice](#)

Namespaces

- [std](#)

6.260.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<valarray>`.

6.261 `gslice_array.h` File Reference

Classes

- class `std::gslice_array<_Tp>`

Namespaces

- `std`

Macros

- `#define _DEFINE_VALARRAY_OPERATOR(_Op, _Name)`

6.261.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<valarray>`.

6.262 `hash_bytes.h` File Reference

Namespaces

- `std`

Functions

- `size_t std::_Fnv_hash_bytes` (const void *__ptr, size_t __len, size_t __seed)
- `size_t std::_Hash_bytes` (const void *__ptr, size_t __len, size_t __seed)

6.262.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<functional>`.

6.263 hash_eq_fn.hpp File Reference

Classes

- struct [__gnu_pbds::detail::hash_eq_fn](#)< Key, Eq_Fn, _Alloc, Store_Hash >
- struct [__gnu_pbds::detail::hash_eq_fn](#)< Key, Eq_Fn, _Alloc, false >
- struct [__gnu_pbds::detail::hash_eq_fn](#)< Key, Eq_Fn, _Alloc, true >

Namespaces

- [__gnu_pbds](#)

6.263.1 Detailed Description

Contains 2 equivalence functions, one employing a hash value, and one ignoring it.

6.264 hash_exponential_size_policy_imp.hpp File Reference

6.264.1 Detailed Description

Contains a resize size policy implementation.

6.265 hash_fun.h File Reference

Namespaces

- [__gnu_cxx](#)

Functions

- `size_t __gnu_cxx::__stl_hash_string(const char *__s)`

6.265.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGI STL subset).

6.266 hash_load_check_resize_trigger_imp.hpp File Reference

Macros

- `#define PB_DS_ASSERT_VALID(X)`
- `#define PB_DS_ASSERT_VALID(X)`

6.266.1 Detailed Description

Contains a resize trigger implementation.

6.267 `hash_load_check_resize_trigger_size_base.hpp` File Reference

Classes

- class [__gnu_pbds::detail::hash_load_check_resize_trigger_size_base< Size_Type, Hold_Size >](#)
- class [__gnu_pbds::detail::hash_load_check_resize_trigger_size_base< Size_Type, true >](#)

Namespaces

- [__gnu_pbds](#)

6.267.1 Detailed Description

Contains an base holding size for some resize policies.

6.268 `hash_map` File Reference

Classes

- class [__gnu_cxx::hash_map< _Key, _Tp, _HashFn, _EqualKey, _Alloc >](#)
- class [__gnu_cxx::hash_multimap< _Key, _Tp, _HashFn, _EqualKey, _Alloc >](#)

Namespaces

- [__gnu_cxx](#)
- [std](#)

Macros

- `#define _BACKWARD_HASH_MAP`

Functions

- `template<class _Key, class _Tp, class _HashFn, class _EqKey, class _Alloc >`
`bool __gnu_cxx::operator!= (const hash_map< _Key, _Tp, _HashFn, _EqKey, _Alloc > &__hm1, const hash_map< _Key, _Tp, _HashFn, _EqKey, _Alloc > &__hm2)`
- `template<class _Key, class _Tp, class _HF, class _EqKey, class _Alloc >`
`bool __gnu_cxx::operator!= (const hash_multimap< _Key, _Tp, _HF, _EqKey, _Alloc > &__hm1, const hash_multimap< _Key, _Tp, _HF, _EqKey, _Alloc > &__hm2)`
- `template<class _Key, class _Tp, class _HashFn, class _EqKey, class _Alloc >`
`bool __gnu_cxx::operator== (const hash_map< _Key, _Tp, _HashFn, _EqKey, _Alloc > &__hm1, const hash_map< _Key, _Tp, _HashFn, _EqKey, _Alloc > &__hm2)`
- `template<class _Key, class _Tp, class _HF, class _EqKey, class _Alloc >`
`bool __gnu_cxx::operator== (const hash_multimap< _Key, _Tp, _HF, _EqKey, _Alloc > &__hm1, const hash_multimap< _Key, _Tp, _HF, _EqKey, _Alloc > &__hm2)`
- `template<class _Key, class _Tp, class _HashFn, class _EqKey, class _Alloc >`
`void __gnu_cxx::swap (hash_map< _Key, _Tp, _HashFn, _EqKey, _Alloc > &__hm1, hash_map< _Key, _Tp, _HashFn, _EqKey, _Alloc > &__hm2)`
- `template<class _Key, class _Tp, class _HashFn, class _EqKey, class _Alloc >`
`void __gnu_cxx::swap (hash_multimap< _Key, _Tp, _HashFn, _EqKey, _Alloc > &__hm1, hash_multimap< _Key, _Tp, _HashFn, _EqKey, _Alloc > &__hm2)`

6.268.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGI STL subset).

6.269 hash_policy.hpp File Reference

Classes

- [class __gnu_pbds::cc_hash_max_collision_check_resize_trigger< External_Load_Access, Size_Type >](#)
- [class __gnu_pbds::direct_mask_range_hashing< Size_Type >](#)
- [class __gnu_pbds::direct_mod_range_hashing< Size_Type >](#)
- [class __gnu_pbds::hash_exponential_size_policy< Size_Type >](#)
- [class __gnu_pbds::hash_load_check_resize_trigger< External_Load_Access, Size_Type >](#)
- [class __gnu_pbds::hash_prime_size_policy](#)
- [class __gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_Type >](#)
- [class __gnu_pbds::linear_probe_fn< Size_Type >](#)
- [class __gnu_pbds::quadratic_probe_fn< Size_Type >](#)

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_SIZE_BASE_C_DEC`

Enumerations

- `enum { num_distinct_sizes_32_bit, num_distinct_sizes_64_bit, num_distinct_sizes }`

Variables

- `static const std::size_t __gnu_pbds::detail::g_a_sizes [num_distinct_sizes_64_bit]`

6.269.1 Detailed Description

Contains hash-related policies.

6.270 hash_prime_size_policy_imp.hpp File Reference

Enumerations

- `enum { num_distinct_sizes_32_bit, num_distinct_sizes_64_bit, num_distinct_sizes }`

Variables

- `static const std::size_t detail::g_a_sizes [num_distinct_sizes_64_bit]`

6.270.1 Detailed Description

Contains a resize size policy implementation.

6.271 hash_set File Reference

Classes

- class [__gnu_cxx::hash_multiset<_Value, _HashFcn, _EqualKey, _Alloc>](#)
- class [__gnu_cxx::hash_set<_Value, _HashFcn, _EqualKey, _Alloc>](#)

Namespaces

- [__gnu_cxx](#)
- [std](#)

Macros

- `#define _BACKWARD_HASH_SET`

Functions

- `template<class _Value, class _HashFcn, class _EqualKey, class _Alloc>
bool __gnu_cxx::operator!= (const hash_set<_Value, _HashFcn, _EqualKey, _Alloc> &__hs1, const hash_set<_Value, _HashFcn, _EqualKey, _Alloc> &__hs2)`
- `template<class _Val, class _HashFcn, class _EqualKey, class _Alloc>
bool __gnu_cxx::operator!= (const hash_multiset<_Val, _HashFcn, _EqualKey, _Alloc> &__hs1, const hash_multiset<_Val, _HashFcn, _EqualKey, _Alloc> &__hs2)`
- `template<class _Value, class _HashFcn, class _EqualKey, class _Alloc>
bool __gnu_cxx::operator== (const hash_set<_Value, _HashFcn, _EqualKey, _Alloc> &__hs1, const hash_set<_Value, _HashFcn, _EqualKey, _Alloc> &__hs2)`
- `template<class _Val, class _HashFcn, class _EqualKey, class _Alloc>
bool __gnu_cxx::operator== (const hash_multiset<_Val, _HashFcn, _EqualKey, _Alloc> &__hs1, const hash_multiset<_Val, _HashFcn, _EqualKey, _Alloc> &__hs2)`
- `template<class _Val, class _HashFcn, class _EqualKey, class _Alloc>
void __gnu_cxx::swap (hash_set<_Val, _HashFcn, _EqualKey, _Alloc> &__hs1, hash_set<_Val, _HashFcn, _EqualKey, _Alloc> &__hs2)`
- `template<class _Val, class _HashFcn, class _EqualKey, class _Alloc>
void __gnu_cxx::swap (hash_multiset<_Val, _HashFcn, _EqualKey, _Alloc> &__hs1, hash_multiset<_Val, _HashFcn, _EqualKey, _Alloc> &__hs2)`

6.271.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGI STL subset).

6.272 `hash_standard_resize_policy_imp.hpp` File Reference

6.272.1 Detailed Description

Contains a resize policy implementation.

6.273 `hashtable.h` File Reference

Classes

- class [std::_Hashtable<_Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits>](#)

Namespaces

- [std](#)

Typedefs

- `template<typename _Tp, typename _Hash >
using std::__cache_default = __not_< __and_< __is_fast_hash< _Hash >, __detail::__is_noexcept_hash< _Tp, _Hash >>>`

6.273.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<unordered_map>` or `<unordered_set>`.

6.274 `hashtable.h` File Reference

Namespaces

- [__gnu_cxx](#)

Enumerations

- enum { **_S_num_primes** }

Functions

- unsigned long **gnu_cxx::__stl_next_prime** (unsigned long __n)
- template<class _Val, class _Key, class _HF, class _Ex, class _Eq, class _All >
bool **gnu_cxx::operator!=** (const hashtable< _Val, _Key, _HF, _Ex, _Eq, _All > &__ht1, const hashtable< _Val, _Key, _HF, _Ex, _Eq, _All > &__ht2)
- template<class _Val, class _Key, class _HF, class _Ex, class _Eq, class _All >
bool **gnu_cxx::operator==** (const hashtable< _Val, _Key, _HF, _Ex, _Eq, _All > &__ht1, const hashtable< _Val, _Key, _HF, _Ex, _Eq, _All > &__ht2)
- template<class _Val, class _Key, class _HF, class _Extract, class _EqKey, class _All >
void **gnu_cxx::swap** (hashtable< _Val, _Key, _HF, _Extract, _EqKey, _All > &__ht1, hashtable< _Val, _Key, _HF, _Extract, _EqKey, _All > &__ht2)

6.274.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGL STL subset).

6.275 hashtable_policy.h File Reference

Classes

- struct [std::__detail::Default_ranged_hash](#)
- struct [std::__detail::Equal_helper< _Key, _Value, _ExtractKey, _Equal, _HashCodeType, __cache_hash_code >](#)
- struct [std::__detail::Equal_helper< _Key, _Value, _ExtractKey, _Equal, _HashCodeType, false >](#)
- struct [std::__detail::Equal_helper< _Key, _Value, _ExtractKey, _Equal, _HashCodeType, true >](#)
- struct [std::__detail::Equality< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, _Unique_keys >](#)
- struct [std::__detail::Equality< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, false >](#)
- struct [std::__detail::Equality< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, true >](#)
- struct [std::__detail::Equality_base](#)
- struct [std::__detail::Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache_hash_code >](#)
- struct [std::__detail::Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, Default_ranged_hash, false >](#)
- struct [std::__detail::Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, Default_ranged_hash, true >](#)
- struct [std::__detail::Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, false >](#)
- struct [std::__detail::Hash_node< _Value, _Cache_hash_code >](#)
- struct [std::__detail::Hash_node< _Value, false >](#)
- struct [std::__detail::Hash_node< _Value, true >](#)
- struct [std::__detail::Hash_node_base](#)
- struct [std::__detail::Hash_node_value_base< _Value >](#)
- struct [std::__detail::Hashtable_alloc< _NodeAlloc >](#)
- struct [std::__detail::Hashtable_alloc< _NodeAlloc >](#)
- struct [std::__detail::Hashtable_base< _Key, _Value, _ExtractKey, _Equal, _H1, _H2, _Hash, _Traits >](#)
- struct [std::__detail::Hashtable_base< _Key, _Value, _ExtractKey, _Equal, _H1, _H2, _Hash, _Traits >](#)
- struct [std::__detail::Hashtable_ebo_helper< _Nm, _Tp, __use_ebo >](#)
- struct [std::__detail::Hashtable_ebo_helper< _Nm, _Tp, false >](#)

- struct `std::__detail::_Hashtable_ebo_helper< _Nm, _Tp, true >`
- struct `std::__detail::_Hashtable_traits< _Cache_hash_code, _Constant_iterators, _Unique_keys >`
- struct `std::__detail::_Insert< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _↵ Traits, _Constant_iterators, _Unique_keys >`
- struct `std::__detail::_Insert< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _↵ Traits, false, _Unique_keys >`
- struct `std::__detail::_Insert< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _↵ Traits, true, false >`
- struct `std::__detail::_Insert< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _↵ Traits, true, true >`
- struct `std::__detail::_Insert_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits >`
- struct `std::__detail::_Local_const_iterator< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __constant_iterators, __cache >`
- struct `std::__detail::_Local_iterator< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __constant_iterators, __cache >`
- struct `std::__detail::_Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache_hash_code >`
- struct `std::__detail::_Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, true >`
- struct `std::__detail::_Map_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, _Unique_keys >`
- struct `std::__detail::_Map_base< _Key, _Pair, _Alloc, _Select1st, _Equal, _H1, _H2, _Hash, _RehashPolicy, _↵ Traits, false >`
- struct `std::__detail::_Map_base< _Key, _Pair, _Alloc, _Select1st, _Equal, _H1, _H2, _Hash, _RehashPolicy, _↵ Traits, true >`
- struct `std::__detail::_Mod_range_hashing`
- struct `std::__detail::_Node_const_iterator< _Value, __constant_iterators, __cache >`
- struct `std::__detail::_Node_iterator< _Value, __constant_iterators, __cache >`
- struct `std::__detail::_Node_iterator_base< _Value, _Cache_hash_code >`
- struct `std::__detail::_Prime_rehash_policy`
- struct `std::__detail::_Rehash_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _Rehash↵ Policy, _Traits >`
- struct `std::__detail::_Rehash_base< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _Prime↵ rehash_policy, _Traits >`
- class `std::_Hashtable< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits >`

Namespaces

- `std`
- `std::__detail`

Typedefs

- template<typename `_Key`, typename `_Value`, typename `_ExtractKey`, typename `_H1`, typename `_H2`, typename `_Hash` >
using **`std::__detail::_hash_code_for_local_iter`** = `_Hash_code_storage< _Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, false >>`

Functions

- `template<class _Iterator >`
`std::iterator_traits< _Iterator >::difference_type std::__detail::__distance_fw (_Iterator __first, _Iterator __last,`
[std::input_iterator_tag](#)`)`
- `template<class _Iterator >`
`std::iterator_traits< _Iterator >::difference_type std::__detail::__distance_fw (_Iterator __first, _Iterator __last,`
[std::forward_iterator_tag](#)`)`
- `template<class _Iterator >`
`std::iterator_traits< _Iterator >::difference_type std::__detail::__distance_fw (_Iterator __first, _Iterator __last)`
- `template<typename _Value , bool _Cache_hash_code>`
`bool std::__detail::operator!= (const _Node_iterator_base< _Value, _Cache_hash_code > &__x, const _Node_iterator_base< _Value, _Cache_hash_code > &__y) noexcept`
- `template<typename _Key , typename _Value , typename _ExtractKey , typename _H1 , typename _H2 , typename _Hash , bool __cache>`
`bool std::__detail::operator!= (const _Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache > &__x, const _Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache > &__y)`
- `template<typename _Value , bool _Cache_hash_code>`
`bool std::__detail::operator== (const _Node_iterator_base< _Value, _Cache_hash_code > &__x, const _Node_iterator_base< _Value, _Cache_hash_code > &__y) noexcept`
- `template<typename _Key , typename _Value , typename _ExtractKey , typename _H1 , typename _H2 , typename _Hash , bool __cache>`
`bool std::__detail::operator== (const _Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache > &__x, const _Local_iterator_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache > &__y)`

6.275.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<unordered_map>` or `<unordered_set>`.

6.276 helper_functions.h File Reference

Namespaces

- [__gnu_debug](#)

Enumerations

- `enum __gnu_debug::Distance_precision { __dp_none, __dp_equality, __dp_sign, __dp_exact }`

Functions

- `template<typename _Iterator >`
`_Iterator __gnu_debug::__base (_Iterator __it)`
- `template<typename _Iterator >`
`_Distance_traits< _Iterator >::__type __gnu_debug::__get_distance (const _Iterator &__lhs, const _Iterator &__rhs, std::random_access_iterator_tag)`
- `template<typename _Iterator >`
`_Distance_traits< _Iterator >::__type __gnu_debug::__get_distance (const _Iterator &__lhs, const _Iterator &__rhs, std::input_iterator_tag)`

- `template<typename _Iterator >`
`_Distance_traits< _Iterator >::__type __gnu_debug::__get_distance (const _Iterator &__lhs, const _Iterator &__rhs)`
- `template<typename _Iterator >`
`_Iterator __gnu_debug::__unsafe (_Iterator __it)`
- `template<typename _InputIterator >`
`bool __gnu_debug::__valid_range (const _InputIterator &__first, const _InputIterator &__last, typename _Distance_traits< _InputIterator >::__type &__dist)`
- `template<typename _InputIterator >`
`bool __gnu_debug::__valid_range (const _InputIterator &__first, const _InputIterator &__last)`
- `template<typename _Integral >`
`bool __gnu_debug::__valid_range_aux (const _Integral &, const _Integral &, typename _Distance_traits< _Integral >::__type &__dist, std::__true_type)`
- `template<typename _InputIterator >`
`bool __gnu_debug::__valid_range_aux (const _InputIterator &__first, const _InputIterator &__last, typename _Distance_traits< _InputIterator >::__type &__dist, std::__false_type)`

6.276.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.277 `indirect_array.h` File Reference

Classes

- class `std::indirect_array< _Tp >`

Namespaces

- `std`

Macros

- `#define _DEFINE_VALARRAY_OPERATOR(_Op, _Name)`

6.277.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<valarray>`.

6.278 `info_fn_imps.hpp` File Reference

6.278.1 Detailed Description

Contains an implementation class for a `binary_heap`.

6.279 info_fn_imps.hpp File Reference

6.279.1 Detailed Description

Contains an implementation class for bin_search_tree_.

6.280 info_fn_imps.hpp File Reference

6.280.1 Detailed Description

Contains implementations of cc_ht_map_'s entire container info related functions.

6.281 info_fn_imps.hpp File Reference

6.281.1 Detailed Description

Contains implementations of gp_ht_map_'s entire container info related functions.

6.282 info_fn_imps.hpp File Reference

6.282.1 Detailed Description

Contains an implementation class for left_child_next_sibling_heap_.

6.283 info_fn_imps.hpp File Reference

6.283.1 Detailed Description

Contains implementations of lu_map_.

6.284 info_fn_imps.hpp File Reference

6.284.1 Detailed Description

Contains an implementation class for ov_tree_.

6.285 `info_fn_imps.hpp` File Reference

6.285.1 Detailed Description

Contains an implementation class for `pat_trie`.

6.286 `info_fn_imps.hpp` File Reference

6.286.1 Detailed Description

Contains an implementation for `rb_tree_`.

6.287 `info_fn_imps.hpp` File Reference

6.287.1 Detailed Description

Contains an implementation.

6.288 `initializer_list` File Reference

Classes

- class `std::initializer_list<_E>`

Namespaces

- `std`

Functions

- `template<class _Tp>`
`constexpr const _Tp * std::begin (initializer_list<_Tp> __ils) noexcept`
- `template<class _Tp>`
`constexpr const _Tp * std::end (initializer_list<_Tp> __ils) noexcept`

6.288.1 Detailed Description

This is a Standard C++ Library header.

6.289 insert_fn_imps.hpp File Reference

6.289.1 Detailed Description

Contains an implementation class for a binary_heap.

6.290 insert_fn_imps.hpp File Reference

6.290.1 Detailed Description

Contains an implementation class for a base of binomial heaps.

6.291 insert_fn_imps.hpp File Reference

6.291.1 Detailed Description

Contains an implementation class for bin_search_tree_.

6.292 insert_fn_imps.hpp File Reference

6.292.1 Detailed Description

Contains implementations of cc_ht_map_'s insert related functions.

6.293 insert_fn_imps.hpp File Reference

6.293.1 Detailed Description

Contains implementations of gp_ht_map_'s insert related functions.

6.294 insert_fn_imps.hpp File Reference

6.294.1 Detailed Description

Contains an implementation class for left_child_next_sibling_heap_.

6.295 insert_fn_imps.hpp File Reference**6.295.1 Detailed Description**

Contains implementations of lu_map_.

6.296 insert_fn_imps.hpp File Reference**6.296.1 Detailed Description**

Contains an implementation class for ov_tree_.

6.297 insert_fn_imps.hpp File Reference**6.297.1 Detailed Description**

Contains an implementation class for a pairing heap.

6.298 insert_fn_imps.hpp File Reference**6.298.1 Detailed Description**

Contains an implementation for rb_tree_.

6.299 insert_fn_imps.hpp File Reference**6.299.1 Detailed Description**

Contains an implementation for rc_binomial_heap_.

6.300 insert_fn_imps.hpp File Reference**6.300.1 Detailed Description**

Contains an implementation class for splay_tree_.

6.301 `insert_fn_imps.hpp` File Reference

6.301.1 Detailed Description

Contains an implementation for `thin_heap_`.

6.302 `insert_join_fn_imps.hpp` File Reference

6.302.1 Detailed Description

Contains an implementation class for `pat_trie`.

6.303 `insert_no_store_hash_fn_imps.hpp` File Reference

6.303.1 Detailed Description

Contains implementations of `cc_ht_map_`'s insert related functions, when the hash value is not stored.

6.304 `insert_no_store_hash_fn_imps.hpp` File Reference

6.304.1 Detailed Description

Contains implementations of `gp_ht_map_`'s insert related functions, when the hash value is not stored.

6.305 `insert_store_hash_fn_imps.hpp` File Reference

6.305.1 Detailed Description

Contains implementations of `cc_ht_map_`'s insert related functions, when the hash value is stored.

6.306 `insert_store_hash_fn_imps.hpp` File Reference

6.306.1 Detailed Description

Contains implementations of `gp_ht_map_`'s find related functions, when the hash value is stored.

6.307 `iomanip` File Reference

Namespaces

- [std](#)

Macros

- `#define __cpp_lib_quoted_string_io`
- `#define _GLIBCXX_IOMANIP`

Functions

- `template<typename _MoneyT >`
`_Get_money< _MoneyT > std::get_money (_MoneyT &__mon, bool __intl=false)`
- `template<typename _CharT >`
`_Get_time< _CharT > std::get_time (std::tm *__tmb, const _CharT *__fmt)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, _↵`
`Resetiosflags __f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, _Setiosflags`
`__f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, _Setbase`
`__f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, _Setfill<`
`_CharT > __f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, _↵`
`Setprecision __f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, _Setw __f)`
- `template<typename _CharT, typename _Traits, typename _MoneyT >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, _Put_↵`
`money< _MoneyT > __f)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, _Put_time<`
`_CharT > __f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__is, _Resetiosflags`
`__f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__is, _Setiosflags`
`__f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__is, _Setbase __f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__is, _Setfill< _↵`
`CharT > __f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__is, _Setprecision`
`__f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__is, _Setw __f)`

- `template<typename _CharT, typename _Traits, typename _MoneyT >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__is, _Get_money<`
`_MoneyT > __f)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__is, _Get_time<`
`_CharT > __f)`
- `template<typename _MoneyT >`
`_Put_money< _MoneyT > std::put_money (const _MoneyT &__mon, bool __intl=false)`
- `template<typename _CharT >`
`_Put_time< _CharT > std::put_time (const std::tm * __tmb, const _CharT * __fmt)`
- `template<typename _CharT >`
`auto std::quoted (const _CharT * __string, _CharT __delim=_CharT(""), _CharT __escape = _CharT("\\"))`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`auto std::quoted (const basic_string< _CharT, _Traits, _Alloc > & __string, _CharT __delim=_CharT(""), _CharT`
`__escape = _CharT("\\"))`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`auto std::quoted (basic_string< _CharT, _Traits, _Alloc > & __string, _CharT __delim=_CharT(""), _CharT __`
`escape = _CharT("\\"))`
- `_Resetiosflags std::resetiosflags (ios_base::fmtflags __mask)`
- `_Setbase std::setbase (int __base)`
- `template<typename _CharT >`
`_Setfill< _CharT > std::setfill (_CharT __c)`
- `_Setiosflags std::setiosflags (ios_base::fmtflags __mask)`
- `_Setprecision std::setprecision (int __n)`
- `_Setw std::setw (int __n)`

6.307.1 Detailed Description

This is a Standard C++ Library header.

6.308 ios File Reference

Macros

- `#define _GLIBCXX_IOS`

6.308.1 Detailed Description

This is a Standard C++ Library header.

6.309 ios_base.h File Reference

Classes

- class `std::ios_base`
- class `std::ios_base::failure`

Namespaces

- [std](#)

Enumerations

- enum `_ios_Fmtflags` {
`_S_boolalpha`, `_S_dec`, `_S_fixed`, `_S_hex`,
`_S_internal`, `_S_left`, `_S_oct`, `_S_right`,
`_S_scientific`, `_S_showbase`, `_S_showpoint`, `_S_showpos`,
`_S_skipws`, `_S_unitbuf`, `_S_uppercase`, `_S_adjustfield`,
`_S_basefield`, `_S_floatfield`, `_S_ios_fmtflags_end`, `_S_ios_fmtflags_max`,
`_S_ios_fmtflags_min` }
- enum `_ios_iostate` {
`_S_goodbit`, `_S_badbit`, `_S_eofbit`, `_S_failbit`,
`_S_ios_iostate_end`, `_S_ios_iostate_max`, `_S_ios_iostate_min` }
- enum `_ios_Openmode` {
`_S_app`, `_S_ate`, `_S_bin`, `_S_in`,
`_S_out`, `_S_trunc`, `_S_ios_openmode_end`, `_S_ios_openmode_max`,
`_S_ios_openmode_min` }
- enum `_ios_Seekdir` { `_S_beg`, `_S_cur`, `_S_end`, `_S_ios_seekdir_end` }
- enum `std::io_errc` { `stream` }

Functions

- `ios_base & std::boolalpha (ios_base & __base)`
- `ios_base & std::dec (ios_base & __base)`
- `ios_base & std::defaultfloat (ios_base & __base)`
- `ios_base & std::fixed (ios_base & __base)`
- `ios_base & std::hex (ios_base & __base)`
- `ios_base & std::hexfloat (ios_base & __base)`
- `ios_base & std::internal (ios_base & __base)`
- `const error_category & std::iostream_category () noexcept`
- `ios_base & std::left (ios_base & __base)`
- `error_code std::make_error_code (io_errc e) noexcept`
- `error_condition std::make_error_condition (io_errc e) noexcept`
- `ios_base & std::noboolalpha (ios_base & __base)`
- `ios_base & std::noshowbase (ios_base & __base)`
- `ios_base & std::noshowpoint (ios_base & __base)`
- `ios_base & std::noshowpos (ios_base & __base)`
- `ios_base & std::noskipws (ios_base & __base)`
- `ios_base & std::nounitbuf (ios_base & __base)`
- `ios_base & std::nouppercase (ios_base & __base)`
- `ios_base & std::oct (ios_base & __base)`
- `constexpr _ios_Fmtflags std::operator& (_ios_Fmtflags __a, _ios_Fmtflags __b)`
- `constexpr _ios_Openmode std::operator& (_ios_Openmode __a, _ios_Openmode __b)`
- `constexpr _ios_iostate std::operator& (_ios_iostate __a, _ios_iostate __b)`
- `const _ios_Fmtflags & std::operator&= (_ios_Fmtflags & __a, _ios_Fmtflags __b)`
- `const _ios_Openmode & std::operator&= (_ios_Openmode & __a, _ios_Openmode __b)`
- `const _ios_iostate & std::operator&= (_ios_iostate & __a, _ios_iostate __b)`

- constexpr `_ios_Fmtflags std::operator^` (`_ios_Fmtflags __a, _ios_Fmtflags __b`)
- constexpr `_ios_Openmode std::operator^` (`_ios_Openmode __a, _ios_Openmode __b`)
- constexpr `_ios_istate std::operator^` (`_ios_istate __a, _ios_istate __b`)
- const `_ios_Fmtflags & std::operator^=` (`_ios_Fmtflags &__a, _ios_Fmtflags __b`)
- const `_ios_Openmode & std::operator^=` (`_ios_Openmode &__a, _ios_Openmode __b`)
- const `_ios_istate & std::operator^=` (`_ios_istate &__a, _ios_istate __b`)
- constexpr `_ios_Fmtflags std::operator|` (`_ios_Fmtflags __a, _ios_Fmtflags __b`)
- constexpr `_ios_Openmode std::operator|` (`_ios_Openmode __a, _ios_Openmode __b`)
- constexpr `_ios_istate std::operator|` (`_ios_istate __a, _ios_istate __b`)
- const `_ios_Fmtflags & std::operator|=` (`_ios_Fmtflags &__a, _ios_Fmtflags __b`)
- const `_ios_Openmode & std::operator|=` (`_ios_Openmode &__a, _ios_Openmode __b`)
- const `_ios_istate & std::operator|=` (`_ios_istate &__a, _ios_istate __b`)
- constexpr `_ios_Fmtflags std::operator~` (`_ios_Fmtflags __a`)
- constexpr `_ios_Openmode std::operator~` (`_ios_Openmode __a`)
- constexpr `_ios_istate std::operator~` (`_ios_istate __a`)
- `ios_base & std::right` (`ios_base &__base`)
- `ios_base & std::scientific` (`ios_base &__base`)
- `ios_base & std::showbase` (`ios_base &__base`)
- `ios_base & std::showpoint` (`ios_base &__base`)
- `ios_base & std::showpos` (`ios_base &__base`)
- `ios_base & std::skipws` (`ios_base &__base`)
- `ios_base & std::unitbuf` (`ios_base &__base`)
- `ios_base & std::uppercase` (`ios_base &__base`)

6.309.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ios>`.

6.310 iosfwd File Reference

Classes

- class `std::basic_filebuf<_CharT, _Traits>`
- class `std::basic_fstream<_CharT, _Traits>`
- class `std::basic_ifstream<_CharT, _Traits>`
- class `std::basic_ios<_CharT, _Traits>`
- class `std::basic_iostream<_CharT, _Traits>`
- class `std::basic_istream<_CharT, _Traits>`
- class `std::basic_istreamstream<_CharT, _Traits, _Alloc>`
- class `std::basic_ofstream<_CharT, _Traits>`
- class `std::basic_ostream<_CharT, _Traits>`
- class `std::basic_ostreamstream<_CharT, _Traits, _Alloc>`
- class `std::basic_streambuf<_CharT, _Traits>`
- class `std::basic_stringbuf<_CharT, _Traits, _Alloc>`
- class `std::basic_stringstream<_CharT, _Traits, _Alloc>`
- class `std::istreambuf_iterator<_CharT, _Traits>`
- class `std::ostreambuf_iterator<_CharT, _Traits>`

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_IOSFWD`

Typedefs

- `typedef basic_filebuf< char > std::filebuf`
- `typedef basic_fstream< char > std::fstream`
- `typedef basic_ifstream< char > std::ifstream`
- `typedef basic_ios< char > std::ios`
- `typedef basic_iostream< char > std::iostream`
- `typedef basic_istream< char > std::istream`
- `typedef basic_istreamstream< char > std::istreamstream`
- `typedef basic_ofstream< char > std::ofstream`
- `typedef basic_ostream< char > std::ostream`
- `typedef basic_ostreamstream< char > std::ostreamstream`
- `typedef basic_streambuf< char > std::streambuf`
- `typedef basic_stringbuf< char > std::stringbuf`
- `typedef basic_stringstream< char > std::stringstream`
- `typedef basic_filebuf< wchar_t > std::wfilebuf`
- `typedef basic_fstream< wchar_t > std::wfstream`
- `typedef basic_ifstream< wchar_t > std::wifstream`
- `typedef basic_ios< wchar_t > std::wios`
- `typedef basic_iostream< wchar_t > std::wiostream`
- `typedef basic_istream< wchar_t > std::wistream`
- `typedef basic_istreamstream< wchar_t > std::wistreamstream`
- `typedef basic_ofstream< wchar_t > std::wofstream`
- `typedef basic_ostream< wchar_t > std::wostream`
- `typedef basic_ostreamstream< wchar_t > std::wostreamstream`
- `typedef basic_streambuf< wchar_t > std::wstreambuf`
- `typedef basic_stringbuf< wchar_t > std::wstringbuf`
- `typedef basic_stringstream< wchar_t > std::wstringstream`

6.310.1 Detailed Description

This is a Standard C++ Library header.

6.311 iostream File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_IOSTREAM`

Variables

- static `ios_base::Init` [std::__ioinit](#)

Standard Stream Objects

The `<iostream>` header declares the eight standard stream objects. For other declarations, see <http://gcc.gnu.org/onlinedocs/libstdc++/manual/io.html> and the [I/O forward declarations](#)

They are required by default to cooperate with the global C library's `FILE` streams, and to be available during program startup and termination. For more information, see the section of the manual linked to above.

- `istream` [std::cin](#)
- `ostream` [std::cout](#)
- `ostream` [std::cerr](#)
- `ostream` [std::clog](#)
- `wistream` [std::wcin](#)
- `wostream` [std::wcout](#)
- `wostream` [std::wcerr](#)
- `wostream` [std::wclog](#)

6.311.1 Detailed Description

This is a Standard C++ Library header.

6.312 istream File Reference

Classes

- class [std::basic_istream<_CharT, _Traits>](#)
- class [std::basic_istream<_CharT, _Traits>](#)
- class [std::basic_istream<_CharT, _Traits>::sentry](#)

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_ISTREAM`

Functions

- `template<typename _CharT, typename _Traits, typename _Tp >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &&__is, _Tp &__x)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::ws (basic_istream< _CharT, _Traits > &__is)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__in, _CharT &__c)`
- `template<class _Traits >`
`basic_istream< char, _Traits > & std::operator>> (basic_istream< char, _Traits > &__in, unsigned char &__c)`
- `template<class _Traits >`
`basic_istream< char, _Traits > & std::operator>> (basic_istream< char, _Traits > &__in, signed char &__c)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__in, _CharT *__s)`
- `template<>`
`basic_istream< char > & std::operator>> (basic_istream< char > &__in, char *__s)`
- `template<class _Traits >`
`basic_istream< char, _Traits > & std::operator>> (basic_istream< char, _Traits > &__in, unsigned char *__s)`
- `template<class _Traits >`
`basic_istream< char, _Traits > & std::operator>> (basic_istream< char, _Traits > &__in, signed char *__s)`

6.312.1 Detailed Description

This is a Standard C++ Library header.

6.313 istream.tcc File Reference

Namespaces

- [std](#)

Macros

- `#define _ISTREAM_TCC`

Functions

- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::ws (basic_istream< _CharT, _Traits > &__is)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__in, _CharT &__c)`
- `template<typename _CharT, typename _Traits >`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__in, _CharT *__s)`

6.313.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<istream>`.

6.314 iterator File Reference

Macros

- `#define _GLIBCXX_ITERATOR`

6.314.1 Detailed Description

This is a Standard C++ Library header.

6.315 iterator File Reference

Namespaces

- [__gnu_cxx](#)

Macros

- `#define _EXT_ITERATOR`

Functions

- `template<typename _InputIterator, typename _Distance >`
`void __gnu_cxx::__distance (_InputIterator __first, _InputIterator __last, _Distance &__n, std::input_iterator_tag)`
- `template<typename _RandomAccessIterator, typename _Distance >`
`void __gnu_cxx::__distance (_RandomAccessIterator __first, _RandomAccessIterator __last, _Distance &__n, std::random_access_iterator_tag)`
- `template<typename _InputIterator, typename _Distance >`
`void __gnu_cxx::distance (_InputIterator __first, _InputIterator __last, _Distance &__n)`

6.315.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGI STL subset).

6.316 iterator File Reference

Classes

- class [std::experimental::fundamentals_v2::ostream_joiner<_DelimT, _CharT, _Traits>](#)

Namespaces

- [std](#)

Macros

- `#define __cpp_lib_experimental_ostream_joiner`
- `#define _GLIBCXX_EXPERIMENTAL_ITERATOR`

Functions

- `template<typename _CharT, typename _Traits, typename _DelimT>
ostream_joiner<decay_t<_DelimT>, _CharT, _Traits> std::experimental::fundamentals_v2::make_ostream_←
_joiner (basic_ostream<_CharT, _Traits> &__os, _DelimT &&__delimiter)`

6.316.1 Detailed Description

This is a TS C++ Library header.

6.316.2 Function Documentation

6.316.2.1 `template<typename _CharT, typename _Traits, typename _DelimT> ostream_joiner<decay_t<_DelimT>, _CharT, _Traits> std::experimental::fundamentals_v2::make_ostream_joiner (basic_ostream<_CharT, _Traits> & __os, _DelimT && __delimiter) [inline]`

Object generator for `ostream_joiner`.

Definition at line 106 of file `experimental/iterator`.

6.317 iterator.h File Reference

Classes

- class [__gnu_parallel::_IteratorPair<_Iterator1, _Iterator2, _IteratorCategory>](#)
- class [__gnu_parallel::_IteratorTriple<_Iterator1, _Iterator2, _Iterator3, _IteratorCategory>](#)

Namespaces

- [__gnu_parallel](#)

6.317.1 Detailed Description

Helper iterator classes for the `std::transform()` functions. This file is a GNU parallel extension to the Standard C++ Library.

6.318 iterator.hpp File Reference

Classes

- class [iterator_](#)

6.318.1 Detailed Description

Contains an `iterator_` class used for ranging over the elements of the table.

6.319 iterator_fn_imps.hpp File Reference

6.319.1 Detailed Description

Contains implementations of `gp_ht_map_`'s iterators related functions, e.g., `begin()`.

6.320 iterator_tracker.h File Reference

Namespaces

- [std](#)
- [std::__profile](#)

Functions

- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`bool std::__profile::operator!= (const __iterator_tracker< _IteratorL, _Sequence > &__lhs, const __iterator_↵`
`tracker< _IteratorR, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`bool std::__profile::operator!= (const __iterator_tracker< _Iterator, _Sequence > &__lhs, const __iterator_↵`
`tracker< _Iterator, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`__iterator_tracker< _Iterator, _Sequence > std::__profile::operator+ (typename __iterator_tracker< _Iterator,`
`_Sequence >::difference_type __n, const __iterator_tracker< _Iterator, _Sequence > &__i) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`__iterator_tracker< _IteratorL, _Sequence >::difference_type std::__profile::operator- (const __iterator_↵`
`tracker< _IteratorL, _Sequence > &__lhs, const __iterator_tracker< _IteratorR, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`__iterator_tracker< _Iterator, _Sequence >::difference_type std::__profile::operator- (const __iterator_↵`
`tracker< _Iterator, _Sequence > &__lhs, const __iterator_tracker< _Iterator, _Sequence > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`bool std::__profile::operator< (const __iterator_tracker< _IteratorL, _Sequence > &__lhs, const __iterator_↵`
`tracker< _IteratorR, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`bool std::__profile::operator< (const __iterator_tracker< _Iterator, _Sequence > &__lhs, const __iterator_↵`
`tracker< _Iterator, _Sequence > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`bool std::__profile::operator<= (const __iterator_tracker< _IteratorL, _Sequence > &__lhs, const __iterator_↵`
`tracker< _IteratorR, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`bool std::__profile::operator<= (const __iterator_tracker< _Iterator, _Sequence > &__lhs, const __iterator_↵`
`tracker< _Iterator, _Sequence > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`bool std::__profile::operator== (const __iterator_tracker< _IteratorL, _Sequence > &__lhs, const __iterator_↵`
`tracker< _IteratorR, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`bool std::__profile::operator== (const __iterator_tracker< _Iterator, _Sequence > &__lhs, const __iterator_↵`
`tracker< _Iterator, _Sequence > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`bool std::__profile::operator> (const __iterator_tracker< _IteratorL, _Sequence > &__lhs, const __iterator_↵`
`tracker< _IteratorR, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`bool std::__profile::operator> (const __iterator_tracker< _Iterator, _Sequence > &__lhs, const __iterator_↵`
`tracker< _Iterator, _Sequence > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`bool std::__profile::operator>= (const __iterator_tracker< _IteratorL, _Sequence > &__lhs, const __iterator_↵`
`tracker< _IteratorR, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`bool std::__profile::operator>= (const __iterator_tracker< _Iterator, _Sequence > &__lhs, const __iterator_↵`
`tracker< _Iterator, _Sequence > &__rhs) noexcept`

6.320.1 Detailed Description

This file is a GNU profile extension to the Standard C++ Library.

6.321 iterators_fn_imps.hpp File Reference

6.321.1 Detailed Description

Contains an implementation class for a binary_heap.

6.322 iterators_fn_imps.hpp File Reference

6.322.1 Detailed Description

Contains an implementation class for bin_search_tree_.

6.323 iterators_fn_imps.hpp File Reference

6.323.1 Detailed Description

Contains implementations of cc_ht_map_'s iterators related functions, e.g., begin().

6.324 iterators_fn_imps.hpp File Reference

6.324.1 Detailed Description

Contains an implementation class for left_child_next_sibling_heap_.

6.325 iterators_fn_imps.hpp File Reference

6.325.1 Detailed Description

Contains implementations of lu_map_.

6.326 iterators_fn_imps.hpp File Reference

6.326.1 Detailed Description

Contains an implementation class for ov_tree_.

6.327 iterators_fn_imps.hpp File Reference

6.327.1 Detailed Description

Contains an implementation class for pat_trie.

6.328 left_child_next_sibling_heap.hpp File Reference

Classes

- class [__gnu_pbds::detail::left_child_next_sibling_heap](#)< Value_Type, Cmp_Fn, Node_Metadata, _Alloc >

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`

6.328.1 Detailed Description

Contains an implementation class for a basic heap.

6.329 lfts_config.h File Reference

6.329.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly.

6.330 limits File Reference

Classes

- struct [std::__numeric_limits_base](#)
- struct [std::numeric_limits< _Tp >](#)
- struct [std::numeric_limits< bool >](#)
- struct [std::numeric_limits< char >](#)
- struct [std::numeric_limits< char16_t >](#)
- struct [std::numeric_limits< char32_t >](#)
- struct [std::numeric_limits< double >](#)
- struct [std::numeric_limits< float >](#)
- struct [std::numeric_limits< int >](#)
- struct [std::numeric_limits< long >](#)
- struct [std::numeric_limits< long double >](#)
- struct [std::numeric_limits< long long >](#)
- struct [std::numeric_limits< short >](#)
- struct [std::numeric_limits< signed char >](#)
- struct [std::numeric_limits< unsigned char >](#)
- struct [std::numeric_limits< unsigned int >](#)
- struct [std::numeric_limits< unsigned long >](#)
- struct [std::numeric_limits< unsigned long long >](#)
- struct [std::numeric_limits< unsigned short >](#)
- struct [std::numeric_limits< wchar_t >](#)

Namespaces

- [std](#)

Macros

- `#define __glibcxx_digits\(T\)`
- `#define __glibcxx_digits10\(T\)`
- `#define __glibcxx_digits10_b\(T, B\)`
- `#define __glibcxx_digits_b\(T, B\)`
- `#define __glibcxx_double_has_denorm_loss`
- `#define __glibcxx_double_tinyness_before`
- `#define __glibcxx_double_traps`
- `#define __glibcxx_float_has_denorm_loss`
- `#define __glibcxx_float_tinyness_before`
- `#define __glibcxx_float_traps`
- `#define __glibcxx_integral_traps`
- `#define __glibcxx_long_double_has_denorm_loss`
- `#define __glibcxx_long_double_tinyness_before`
- `#define __glibcxx_long_double_traps`
- `#define __glibcxx_max\(T\)`
- `#define __glibcxx_max_b\(T, B\)`
- `#define __glibcxx_max_digits10\(T\)`
- `#define __glibcxx_min\(T\)`

- `#define __glibcxx_min_b(T, B)`
- `#define __glibcxx_signed(T)`
- `#define __glibcxx_signed_b(T, B)`
- `#define __INT_N(TYPE, BITSIZE, EXT, UEXT)`
- `#define __INT_N_201103(TYPE)`
- `#define __INT_N_U201103(TYPE)`
- `#define _GLIBCXX_NUMERIC_LIMITS`

Enumerations

- enum `std::float_denorm_style` { `std::denorm_indeterminate`, `std::denorm_absent`, `std::denorm_present` }
- enum `std::float_round_style` { `round_indeterminate`, `std::round_toward_zero`, `std::round_to_nearest`, `std::round_toward_infinity`, `std::round_toward_neg_infinity` }

6.330.1 Detailed Description

This is a Standard C++ Library header.

6.331 linear_probe_fn_imp.hpp File Reference

6.331.1 Detailed Description

Contains a probe policy implementation

6.332 list File Reference

Macros

- `#define _GLIBCXX_LIST`

6.332.1 Detailed Description

This is a Standard C++ Library header.

6.333 list File Reference

Classes

- class `std::__debug::list< _Tp, _Allocator >`

Namespaces

- [__gnu_debug](#)
- [std](#)
- [std::__debug](#)

Macros

- `#define _GLIBCXX_DEBUG_LIST`

Functions

- `template<typename _Tp, typename _Alloc >`
`bool std::__debug::operator!= (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool std::__debug::operator< (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool std::__debug::operator<= (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool std::__debug::operator== (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool std::__debug::operator> (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool std::__debug::operator>= (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`void std::__debug::swap (list< _Tp, _Alloc > &__lhs, list< _Tp, _Alloc > &__rhs) noexcept(/*conditional */)`

6.333.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.334 list File Reference

Classes

- class [std::__profile::list< _Tp, _Allocator >](#)

Namespaces

- [std](#)
- [std::__profile](#)

Macros

- `#define _GLIBCXX_PROFILE_LIST`

Functions

- `template<typename _Tp, typename _Alloc >`
`bool std::__profile::operator!= (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool std::__profile::operator< (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool std::__profile::operator<= (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool std::__profile::operator== (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool std::__profile::operator> (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`bool std::__profile::operator>= (const list< _Tp, _Alloc > &__lhs, const list< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >`
`void std::__profile::swap (list< _Tp, _Alloc > &__lhs, list< _Tp, _Alloc > &__rhs) noexcept(/*conditional */)`

6.334.1 Detailed Description

This file is a GNU profile extension to the Standard C++ Library.

6.335 list File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_LIST`

Typedefs

- `template<typename _Tp >`
`using std::experimental::fundamentals_v2::pmr::list = std::list< _Tp, polymorphic_allocator< _Tp >>`

Functions

- `template<typename _Tp, typename _Alloc, typename _Up >`
`void std::experimental::fundamentals_v2::erase (list< _Tp, _Alloc > &__cont, const _Up &__value)`
- `template<typename _Tp, typename _Alloc, typename _Predicate >`
`void std::experimental::fundamentals_v2::erase_if (list< _Tp, _Alloc > &__cont, _Predicate __pred)`

6.335.1 Detailed Description

This is a TS C++ Library header.

6.336 list.tcc File Reference

Namespaces

- [std](#)

Macros

- `#define _LIST_TCC`

6.336.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<list>`.

6.337 list_partition.h File Reference

Namespaces

- [__gnu_parallel](#)

Functions

- `template<typename _Iter >`
`void __gnu_parallel::__shrink (std::vector< _Iter > &__os_starts, size_t &__count_to_two, size_t &__range_↵`
`length)`
- `template<typename _Iter >`
`void __gnu_parallel::__shrink_and_double (std::vector< _Iter > &__os_starts, size_t &__count_to_two, size_t`
`&__range_length, const bool __make_twice)`
- `template<typename _Iter, typename _FunctorType >`
`size_t __gnu_parallel::list_partition (const _Iter __begin, const _Iter __end, _Iter *__starts, size_t *__lengths,`
`const int __num_parts, _FunctorType &__f, int __oversampling=0)`

6.337.1 Detailed Description

__Functionality to split __sequence referenced by only input iterators. This file is a GNU parallel extension to the Standard C++ Library.

6.338 list_update_policy.hpp File Reference

Classes

- class [__gnu_pbds::lu_counter_policy< Max_Count, _Alloc >](#)
- class [__gnu_pbds::lu_move_to_front_policy< _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

6.338.1 Detailed Description

Contains policies for list update containers.

6.339 locale File Reference

Macros

- `#define _GLIBCXX_LOCALE`

6.339.1 Detailed Description

This is a Standard C++ Library header.

6.340 locale_classes.h File Reference

Classes

- class [std::collate<_CharT>](#)
- class [std::collate_byname<_CharT>](#)
- class [std::locale](#)
- class [std::locale::facet](#)
- class [std::locale::id](#)

Namespaces

- [std](#)

6.340.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

6.341 locale_classes.tcc File Reference

Namespaces

- [std](#)

Macros

- `#define _LOCALE_CLASSES_TCC`

Functions

- `template<typename _Facet >`
`bool std::has_facet (const locale &__loc) throw ()`
- `template<typename _Facet >`
`const _Facet & std::use_facet (const locale &__loc)`

6.341.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

6.342 locale_conv.h File Reference

Classes

- class `std::wbuffer_convert< _Codecvt, _Elem, _Tr >`
- class `std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _Byte_alloc >`

Namespaces

- `std`

Functions

- `template<typename _OutStr, typename _InChar, typename _Codecvt, typename _State, typename _Fn >`
`bool std::__do_str_codecvt (const _InChar *__first, const _InChar *__last, _OutStr &__outstr, const _Codecvt &__cvt, _State &__state, size_t &__count, _Fn __fn)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool std::__str_codecvt_in (const char *__first, const char *__last, basic_string< _CharT, _Traits, _Alloc > &__outstr, const codecvt< _CharT, char, _State > &__cvt, _State &__state, size_t &__count)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool std::__str_codecvt_in (const char *__first, const char *__last, basic_string< _CharT, _Traits, _Alloc > &__outstr, const codecvt< _CharT, char, _State > &__cvt)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool std::__str_codecvt_out (const _CharT *__first, const _CharT *__last, basic_string< char, _Traits, _Alloc > &__outstr, const codecvt< _CharT, char, _State > &__cvt, _State &__state, size_t &__count)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _State >`
`bool std::__str_codecvt_out (const _CharT *__first, const _CharT *__last, basic_string< char, _Traits, _Alloc > &__outstr, const codecvt< _CharT, char, _State > &__cvt)`

6.342.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

6.343 `locale_facets.h` File Reference

Classes

- class [std::__ctype_abstract_base<_CharT>](#)
- class [std::ctype<_CharT>](#)
- class [std::ctype<char>](#)
- class [std::ctype<wchar_t>](#)
- class [std::ctype_byname<_CharT>](#)
- class [std::ctype_byname<char>](#)
- class [std::num_get<_CharT, _InIter>](#)
- class [std::num_put<_CharT, _OutIter>](#)
- class [std::num_punct<_CharT>](#)
- class [std::num_punct_byname<_CharT>](#)

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_NUM_CXX11_FACETS`
- `#define _GLIBCXX_NUM_FACETS`
- `#define _GLIBCXX_NUM_UNICODE_FACETS`

Functions

- `template<typename _CharT>`
`_CharT * std::__add_grouping (_CharT *__s, _CharT __sep, const char *__gbeg, size_t __gsize, const _CharT`
`* __first, const _CharT * __last)`
- `template<typename _Tp>`
`void std::__convert_to_v (const char *, _Tp &, ios_base::iostate &, const __c_locale &) throw ()`
- `template<>`
`void std::__convert_to_v (const char *, float &, ios_base::iostate &, const __c_locale &) throw ()`
- `template<>`
`void std::__convert_to_v (const char *, double &, ios_base::iostate &, const __c_locale &) throw ()`
- `template<>`
`void std::__convert_to_v (const char *, long double &, ios_base::iostate &, const __c_locale &) throw ()`
- `template<typename _CharT>`
`ostreambuf_iterator<_CharT> std::__write (ostreambuf_iterator<_CharT> __s, const _CharT * __ws, int`
`__len)`

- `template<typename _CharT, typename _OutIter >
_OutIter std::write (_OutIter __s, const _CharT *__ws, int __len)`
- `template<typename _CharT >
bool std::isalnum (_CharT __c, const locale &__loc)`
- `template<typename _CharT >
bool std::isalpha (_CharT __c, const locale &__loc)`
- `template<typename _CharT >
bool std::isblank (_CharT __c, const locale &__loc)`
- `template<typename _CharT >
bool std::iscntrl (_CharT __c, const locale &__loc)`
- `template<typename _CharT >
bool std::isdigit (_CharT __c, const locale &__loc)`
- `template<typename _CharT >
bool std::isgraph (_CharT __c, const locale &__loc)`
- `template<typename _CharT >
bool std::islower (_CharT __c, const locale &__loc)`
- `template<typename _CharT >
bool std::isprint (_CharT __c, const locale &__loc)`
- `template<typename _CharT >
bool std::ispunct (_CharT __c, const locale &__loc)`
- `template<typename _CharT >
bool std::isspace (_CharT __c, const locale &__loc)`
- `template<typename _CharT >
bool std::isupper (_CharT __c, const locale &__loc)`
- `template<typename _CharT >
bool std::isxdigit (_CharT __c, const locale &__loc)`
- `template<typename _CharT >
_CharT std::tolower (_CharT __c, const locale &__loc)`
- `template<typename _CharT >
_CharT std::toupper (_CharT __c, const locale &__loc)`

6.343.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

6.344 locale_facets.tcc File Reference

Namespaces

- `std`

Macros

- `#define _LOCALE_FACETS_TCC`

Functions

- `template<typename _CharT >`
`_CharT * std::__add_grouping (_CharT *__s, _CharT __sep, const char *__gbeg, size_t __gsize, const _CharT`
`* __first, const _CharT * __last)`
- `template<typename _CharT, typename _ValueT >`
`int std::__int_to_char (_CharT *__bufend, _ValueT __v, const _CharT * __lit, ios_base::fmtflags __flags, bool`
`__dec)`
- `bool std::__verify_grouping (const char *__grouping, size_t __grouping_size, const string & __grouping_tmp)`
`throw ()`

6.344.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

6.345 locale_facets_nonio.h File Reference

Classes

- class `std::messages< _CharT >`
- struct `std::messages_base`
- class `std::messages_byname< _CharT >`
- class `std::money_base`
- class `std::money_get< _CharT, _InIter >`
- class `std::money_put< _CharT, _OutIter >`
- class `std::moneypunct< _CharT, _Intl >`
- class `std::moneypunct_byname< _CharT, _Intl >`
- class `std::time_base`
- class `std::time_get< _CharT, _InIter >`
- class `std::time_get_byname< _CharT, _InIter >`
- class `std::time_put< _CharT, _OutIter >`
- class `std::time_put_byname< _CharT, _OutIter >`

Namespaces

- `std`

6.345.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

6.346 locale_facets_nonio.tcc File Reference

Namespaces

- [std](#)

Macros

- `#define _LOCALE_FACETS_NONIO_TCC`

6.346.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

6.347 localefwd.h File Reference

Classes

- class [std::codecvt<_InternT, _ExternT, _StateT>](#)
- class [std::codecvt_byname<_InternT, _ExternT, _StateT>](#)
- class [std::collate<_CharT>](#)
- class [std::collate_byname<_CharT>](#)
- class [std::ctype<_CharT>](#)
- class [std::ctype_byname<_CharT>](#)
- class [std::messages<_CharT>](#)
- class [std::messages_byname<_CharT>](#)
- class [std::money_get<_CharT, _InIter>](#)
- class [std::money_put<_CharT, _OutIter>](#)
- class [std::moneypunct<_CharT, _Intl>](#)
- class [std::moneypunct_byname<_CharT, _Intl>](#)
- class [std::num_get<_CharT, _InIter>](#)
- class [std::num_put<_CharT, _OutIter>](#)
- class [std::numpunct<_CharT>](#)
- class [std::numpunct_byname<_CharT>](#)
- class [std::time_get<_CharT, _InIter>](#)
- class [std::time_get_byname<_CharT, _InIter>](#)
- class [std::time_put<_CharT, _OutIter>](#)
- class [std::time_put_byname<_CharT, _OutIter>](#)

Namespaces

- [std](#)

Functions

- `template<typename _Facet >`
`bool std::has_facet (const locale &__loc) throw ()`
- `template<typename _CharT >`
`bool std::isalnum (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::isalpha (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::isblank (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::iscntrl (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::isdigit (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::isgraph (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::islower (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::isprint (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::ispunct (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::isspace (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::isupper (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`bool std::isxdigit (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`_CharT std::tolower (_CharT __c, const locale &__loc)`
- `template<typename _CharT >`
`_CharT std::toupper (_CharT __c, const locale &__loc)`
- `template<typename _Facet >`
`const _Facet & std::use_facet (const locale &__loc)`

6.347.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

6.348 `losertree.h` File Reference

Classes

- `class __gnu_parallel::LoserTree< __stable, _Tp, _Compare >`
- `class __gnu_parallel::LoserTree< false, _Tp, _Compare >`
- `class __gnu_parallel::LoserTreeBase< _Tp, _Compare >`
- `struct __gnu_parallel::LoserTreeBase< _Tp, _Compare >::Loser`
- `class __gnu_parallel::LoserTreePointer< __stable, _Tp, _Compare >`

- class [__gnu_parallel::_LoserTreePointer< false, _Tp, _Compare >](#)
- class [__gnu_parallel::_LoserTreePointerBase< _Tp, _Compare >](#)
- struct [__gnu_parallel::_LoserTreePointerBase< _Tp, _Compare >::_Loser](#)
- class [__gnu_parallel::_LoserTreePointerUnguarded< __stable, _Tp, _Compare >](#)
- class [__gnu_parallel::_LoserTreePointerUnguarded< false, _Tp, _Compare >](#)
- class [__gnu_parallel::_LoserTreePointerUnguardedBase< _Tp, _Compare >](#)
- class [__gnu_parallel::_LoserTreeUnguarded< __stable, _Tp, _Compare >](#)
- class [__gnu_parallel::_LoserTreeUnguarded< false, _Tp, _Compare >](#)
- class [__gnu_parallel::_LoserTreeUnguardedBase< _Tp, _Compare >](#)

Namespaces

- [__gnu_parallel](#)

6.348.1 Detailed Description

Many generic loser tree variants. This file is a GNU parallel extension to the Standard C++ Library.

6.349 lu_counter_metadata.hpp File Reference

Classes

- class [__gnu_pbds::detail::lu_counter_metadata< Size_Type >](#)
- class [__gnu_pbds::detail::lu_counter_policy_base< Size_Type >](#)
- class [__gnu_pbds::detail::lu_counter_policy_base< Size_Type >](#)

Namespaces

- [__gnu_pbds](#)

6.349.1 Detailed Description

Contains implementation of a lu counter policy's metadata.

6.350 lu_map.hpp File Reference

Classes

- class [__gnu_pbds::detail::lu_map< Key, Mapped, Eq_Fn, _Alloc, Update_Policy >](#)

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_GEN_POS`
- `#define PB_DS_LU_NAME`
- `#define PB_DS_LU_TRAITS_BASE`

6.350.1 Detailed Description

Contains a list update map.

6.351 macros.h File Reference

Macros

- `#define __glibcxx_check_bucket_index(_N)`
- `#define __glibcxx_check_equal_allocs(_This, _Other)`
- `#define __glibcxx_check_erase(_Position)`
- `#define __glibcxx_check_erase_after(_Position)`
- `#define __glibcxx_check_erase_range(_First, _Last)`
- `#define __glibcxx_check_erase_range_after(_First, _Last)`
- `#define __glibcxx_check_heap(_First, _Last)`
- `#define __glibcxx_check_heap_pred(_First, _Last, _Pred)`
- `#define __glibcxx_check_insert(_Position)`
- `#define __glibcxx_check_insert_after(_Position)`
- `#define __glibcxx_check_insert_range(_Position, _First, _Last, _Dist)`
- `#define __glibcxx_check_insert_range_after(_Position, _First, _Last, _Dist)`
- `#define __glibcxx_check_irreflexive(_First, _Last)`
- `#define __glibcxx_check_irreflexive2(_First, _Last)`
- `#define __glibcxx_check_irreflexive_pred(_First, _Last, _Pred)`
- `#define __glibcxx_check_irreflexive_pred2(_First, _Last, _Pred)`
- `#define __glibcxx_check_max_load_factor(_F)`
- `#define __glibcxx_check_non_empty_range(_First, _Last)`
- `#define __glibcxx_check_nonempty()`
- `#define __glibcxx_check_partitioned_lower(_First, _Last, _Value)`
- `#define __glibcxx_check_partitioned_lower_pred(_First, _Last, _Value, _Pred)`
- `#define __glibcxx_check_partitioned_upper(_First, _Last, _Value)`
- `#define __glibcxx_check_partitioned_upper_pred(_First, _Last, _Value, _Pred)`
- `#define __glibcxx_check_self_move_assign(_Other)`
- `#define __glibcxx_check_sorted(_First, _Last)`
- `#define __glibcxx_check_sorted_pred(_First, _Last, _Pred)`
- `#define __glibcxx_check_sorted_set(_First1, _Last1, _First2)`
- `#define __glibcxx_check_sorted_set_pred(_First1, _Last1, _First2, _Pred)`
- `#define __glibcxx_check_string(_String)`
- `#define __glibcxx_check_string_len(_String, _Len)`
- `#define __glibcxx_check_subscript(_N)`
- `#define __glibcxx_check_valid_range(_First, _Last)`
- `#define __glibcxx_check_valid_range2(_First, _Last, _Dist)`
- `#define __GLIBCXX_DEBUG_VERIFY(_Condition, _ErrorMessage)`
- `#define __GLIBCXX_DEBUG_VERIFY_AT(_Condition, _ErrorMessage, _File, _Line)`

6.351.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.351.2 Macro Definition Documentation

6.351.2.1 `#define __glibcxx_check_erase(_Position)`

Verify that we can erase the element referenced by the iterator `_Position`. We can erase the element if the `_Position` iterator is dereferenceable and references this sequence.

Definition at line 145 of file macros.h.

6.351.2.2 `#define __glibcxx_check_erase_after(_Position)`

Verify that we can erase the element after the iterator `_Position`. We can erase the element if the `_Position` iterator is before a dereferenceable one and references this sequence.

Definition at line 159 of file macros.h.

6.351.2.3 `#define __glibcxx_check_erase_range(_First, _Last)`

Verify that we can erase the elements in the iterator range `[_First, _Last)`. We can erase the elements if `[_First, _Last)` is a valid iterator range within this sequence.

Definition at line 173 of file macros.h.

6.351.2.4 `#define __glibcxx_check_erase_range_after(_First, _Last)`

Verify that we can erase the elements in the iterator range `(_First, _Last)`. We can erase the elements if `(_First, _Last)` is a valid iterator range within this sequence.

Definition at line 185 of file macros.h.

6.351.2.5 `#define __glibcxx_check_heap_pred(_First, _Last, _Pred)`

Verify that the iterator range `[_First, _Last)` is a heap w.r.t. the predicate `_Pred`.

Definition at line 335 of file macros.h.

6.351.2.6 `#define __glibcxx_check_insert(_Position)`

Verify that we can insert into `*this` with the iterator `_Position`. Insertion into a container at a specific position requires that the iterator be nonsingular, either dereferenceable or past-the-end, and that it reference the sequence we are inserting into. Note that this macro is only valid when the container is a `_Safe_sequence` and the iterator is a `_Safe_iterator`.

Definition at line 79 of file macros.h.

6.351.2.7 `#define __glibcxx_check_insert_after(_Position)`

Verify that we can insert into `*this` after the iterator `_Position`. Insertion into a container after a specific position requires that the iterator be nonsingular, either dereferenceable or before-begin, and that it reference the sequence we are inserting into. Note that this macro is only valid when the container is a `_Safe_sequence` and the iterator is a `_Safe_iterator`.

Definition at line 96 of file `macros.h`.

6.351.2.8 `#define __glibcxx_check_insert_range(_Position, _First, _Last, _Dist)`

Verify that we can insert the values in the iterator range `[_First, _Last)` into `*this` with the iterator `_Position`. Insertion into a container at a specific position requires that the iterator be nonsingular (i.e., either dereferenceable or past-the-end), that it reference the sequence we are inserting into, and that the iterator range `[_First, _Last)` is a valid (possibly empty) range which does not reference the sequence we are inserting into. Note that this macro is only valid when the container is a `_Safe_sequence` and the `_Position` iterator is a `_Safe_iterator`.

Definition at line 113 of file `macros.h`.

6.351.2.9 `#define __glibcxx_check_insert_range_after(_Position, _First, _Last, _Dist)`

Verify that we can insert the values in the iterator range `[_First, _Last)` into `*this` after the iterator `_Position`. Insertion into a container after a specific position requires that the iterator be nonsingular (i.e., either dereferenceable or past-the-end), that it reference the sequence we are inserting into, and that the iterator range `[_First, _Last)` is a valid (possibly empty) range which does not reference the sequence we are inserting into. Note that this macro is only valid when the container is a `_Safe_sequence` and the `_Position` iterator is a `_Safe_iterator`.

Definition at line 132 of file `macros.h`.

6.351.2.10 `#define __glibcxx_check_partitioned_lower(_First, _Last, _Value)`

Verify that the iterator range `[_First, _Last)` is partitioned w.r.t. the value `_Value`.

Definition at line 279 of file `macros.h`.

6.351.2.11 `#define __glibcxx_check_partitioned_lower_pred(_First, _Last, _Value, _Pred)`

Verify that the iterator range `[_First, _Last)` is partitioned w.r.t. the value `_Value` and predicate `_Pred`.

Definition at line 301 of file `macros.h`.

6.351.2.12 `#define __glibcxx_check_partitioned_upper_pred(_First, _Last, _Value, _Pred)`

Verify that the iterator range `[_First, _Last)` is partitioned w.r.t. the value `_Value` and predicate `_Pred`.

Definition at line 314 of file `macros.h`.

6.351.2.13 `#define __glibcxx_check_sorted_pred(_First, _Last, _Pred)`

Verify that the iterator range `[_First, _Last)` is sorted by the predicate `_Pred`.

Definition at line 245 of file `macros.h`.

6.351.2.14 `#define _GLIBCXX_DEBUG_VERIFY_AT(_Condition, _ErrorMessage, _File, _Line)`

Macros used by the implementation to verify certain properties. These macros may only be used directly by the debug wrappers. Note that these are macros (instead of the more obviously *correct* choice of making them functions) because we need line and file information at the call site, to minimize the distance between the user error and where the error is reported.

Definition at line 41 of file `macros.h`.

6.352 malloc_allocator.h File Reference

Classes

- class [__gnu_cxx::malloc_allocator< _Tp >](#)

Namespaces

- [__gnu_cxx](#)

Functions

- `template<typename _Tp >`
`bool __gnu_cxx::operator!= (const malloc_allocator< _Tp > &, const malloc_allocator< _Tp > &)`
- `template<typename _Tp >`
`bool __gnu_cxx::operator== (const malloc_allocator< _Tp > &, const malloc_allocator< _Tp > &)`

6.352.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.353 map File Reference

Macros

- `#define _GLIBCXX_MAP`

6.353.1 Detailed Description

This is a Standard C++ Library header.

6.354 map File Reference

Macros

- `#define _GLIBCXX_DEBUG_MAP`

6.354.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.355 map File Reference

Macros

- `#define _GLIBCXX_PROFILE_MAP`

6.355.1 Detailed Description

This file is a GNU profile extension to the Standard C++ Library.

6.356 map File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_MAP`

Typedefs

- `template<typename _Key, typename _Tp, typename _Compare = less<_Key>>>`
`using std::experimental::fundamentals_v2::pmr::map = std::map< _Key, _Tp, _Compare, polymorphic_↵`
`allocator< pair< const _Key, _Tp >>>`
- `template<typename _Key, typename _Tp, typename _Compare = less<_Key>>>`
`using std::experimental::fundamentals_v2::pmr::multimap = std::multimap< _Key, _Tp, _Compare,`
`polymorphic_allocator< pair< const _Key, _Tp >>>`

Functions

- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc, typename _Predicate >`
`void std::experimental::fundamentals_v2::erase_if (map< _Key, _Tp, _Compare, _Alloc > &__cont, _↔`
`Predicate __pred)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc, typename _Predicate >`
`void std::experimental::fundamentals_v2::erase_if (multimap< _Key, _Tp, _Compare, _Alloc > &__cont, _↔`
`Predicate __pred)`

6.356.1 Detailed Description

This is a TS C++ Library header.

6.357 map.h File Reference

Classes

- class `std::__debug::map< _Key, _Tp, _Compare, _Allocator >`

Namespaces

- `std`
- `std::__debug`

Functions

- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool std::__debug::operator!= (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key,`
`_Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool std::__debug::operator< (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key,`
`_Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool std::__debug::operator<= (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key,`
`_Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool std::__debug::operator== (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key,`
`_Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool std::__debug::operator> (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key,`
`_Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`bool std::__debug::operator>= (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key,`
`_Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >`
`void std::__debug::swap (map< _Key, _Tp, _Compare, _Allocator > &__lhs, map< _Key, _Tp, _Compare,`
`_Allocator > &__rhs) noexcept(/*conditional */)`

6.357.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.358 map.h File Reference

Classes

- class [std::__profile::map< _Key, _Tp, _Compare, _Allocator >](#)

Namespaces

- [std](#)
- [std::__profile](#)

Functions

- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >
bool std::__profile::operator!= (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >
bool std::__profile::operator< (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >
bool std::__profile::operator<= (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >
bool std::__profile::operator== (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >
bool std::__profile::operator> (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >
bool std::__profile::operator>= (const map< _Key, _Tp, _Compare, _Allocator > &__lhs, const map< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >
void std::__profile::swap (map< _Key, _Tp, _Compare, _Allocator > &__lhs, map< _Key, _Tp, _Compare, _Allocator > &__rhs) noexcept(/*conditional */)`

6.358.1 Detailed Description

This file is a GNU profile extension to the Standard C++ Library.

6.359 mask_array.h File Reference

Classes

- class [std::mask_array< _Tp >](#)

Namespaces

- [std](#)

Macros

- `#define _DEFINE_VALARRAY_OPERATOR(_Op, _Name)`

6.359.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<valarray>`.

6.360 mask_based_range_hashing.hpp File Reference

Classes

- class [__gnu_pbds::detail::mask_based_range_hashing< Size_Type >](#)

Namespaces

- [__gnu_pbds](#)

6.360.1 Detailed Description

Contains a range hashing policy base.

6.361 math.h File Reference

6.361.1 Detailed Description

This is a Standard C++ Library header.

6.362 memory File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_MEMORY`

Enumerations

- enum `pointer_safety` { `relaxed`, `preferred`, `strict` }

Functions

- void * `std::align` (size_t __align, size_t __size, void *&__ptr, size_t &__space) noexcept
- void `std::declare_no_pointers` (char *, size_t)
- void `std::declare_reachable` (void *)
- pointer_safety `std::get_pointer_safety` () noexcept
- void `std::undeclare_no_pointers` (char *, size_t)
- template<class T >
T * `std::undeclare_reachable` (T *__p)

6.362.1 Detailed Description

This is a Standard C++ Library header.

6.363 memory File Reference

Classes

- struct `__gnu_cxx::temporary_buffer<_ForwardIterator, _Tp>`

Namespaces

- `__gnu_cxx`

Macros

- `#define _EXT_MEMORY`

Functions

- `template<typename _InputIter, typename _Size, typename _ForwardIter >`
`pair< _InputIter, _ForwardIter > __gnu_cxx::__uninitialized_copy_n (_InputIter __first, _Size __count, _↵`
`ForwardIter __result, std::input_iterator_tag)`
- `template<typename _RandomAccessIter, typename _Size, typename _ForwardIter >`
`pair< _RandomAccessIter, _ForwardIter > __gnu_cxx::__uninitialized_copy_n (_RandomAccessIter __first,`
`_Size __count, _ForwardIter __result, std::random_access_iterator_tag)`
- `template<typename _InputIter, typename _Size, typename _ForwardIter >`
`pair< _InputIter, _ForwardIter > __gnu_cxx::__uninitialized_copy_n (_InputIter __first, _Size __count, _↵`
`ForwardIter __result)`
- `template<typename _InputIter, typename _Size, typename _ForwardIter, typename _Allocator >`
`pair< _InputIter, _ForwardIter > __gnu_cxx::__uninitialized_copy_n_a (_InputIter __first, _Size __count, _↵`
`ForwardIter __result, _Allocator __alloc)`
- `template<typename _InputIter, typename _Size, typename _ForwardIter, typename _Tp >`
`pair< _InputIter, _ForwardIter > __gnu_cxx::__uninitialized_copy_n_a (_InputIter __first, _Size __count, _↵`
`ForwardIter __result, std::allocator< _Tp >)`
- `template<typename _InputIter, typename _Size, typename _ForwardIter >`
`pair< _InputIter, _ForwardIter > __gnu_cxx::uninitialized_copy_n (_InputIter __first, _Size __count, _ForwardIter`
`__result)`

6.363.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGL STL subset).

6.364 memory File Reference

Namespaces

- [std](#)

Macros

- `#define __cpp_lib_experimental_observer_ptr`
- `#define _GLIBCXX_EXPERIMENTAL_MEMORY`

Functions

- `template<typename _Tp >`
`observer_ptr< _Tp > std::experimental::fundamentals_v2::make_observer (_Tp *__p) noexcept`
- `template<typename _Tp, typename _Up >`
`bool std::experimental::fundamentals_v2::operator!= (observer_ptr< _Tp > __p1, observer_ptr< _Up > _↵`
`__p2)`
- `template<typename _Tp >`
`bool std::experimental::fundamentals_v2::operator!= (observer_ptr< _Tp > __p, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool std::experimental::fundamentals_v2::operator!= (nullptr_t, observer_ptr< _Tp > __p) noexcept`

- `template<typename _Tp, typename _Up >`
`bool std::experimental::fundamentals_v2::operator< (observer_ptr< _Tp > __p1, observer_ptr< _Up > __p2)`
- `template<typename _Tp, typename _Up >`
`bool std::experimental::fundamentals_v2::operator<= (observer_ptr< _Tp > __p1, observer_ptr< _Up > __p2)`
- `template<typename _Tp, typename _Up >`
`bool std::experimental::fundamentals_v2::operator== (observer_ptr< _Tp > __p1, observer_ptr< _Up > __p2)`
- `template<typename _Tp >`
`bool std::experimental::fundamentals_v2::operator== (observer_ptr< _Tp > __p, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool std::experimental::fundamentals_v2::operator== (nullptr_t, observer_ptr< _Tp > __p) noexcept`
- `template<typename _Tp, typename _Up >`
`bool std::experimental::fundamentals_v2::operator> (observer_ptr< _Tp > __p1, observer_ptr< _Up > __p2)`
- `template<typename _Tp, typename _Up >`
`bool std::experimental::fundamentals_v2::operator>= (observer_ptr< _Tp > __p1, observer_ptr< _Up > __p2)`
- `template<typename _Tp >`
`void std::experimental::fundamentals_v2::swap (observer_ptr< _Tp > &__p1, observer_ptr< _Tp > &__p2) noexcept`

6.364.1 Detailed Description

This is a TS C++ Library header.

6.365 memory_resource File Reference

Namespaces

- [std](#)

Macros

- `#define __cpp_lib_experimental_memory_resources`
- `#define _GLIBCXX_EXPERIMENTAL_MEMORY_RESOURCE`

Typedefs

- `template<typename _Alloc >`
`using std::experimental::fundamentals_v2::pmr::resource_adaptor = __resource_adaptor_imp< typename allocator_traits< _Alloc >::template rebind_alloc< char >>`

Functions

- [std::atomic](#)< memory_resource * > & [std::experimental::fundamentals_v2::pmr::__get_default_resource](#) ()
- memory_resource * [std::experimental::fundamentals_v2::pmr::get_default_resource](#) () noexcept
- memory_resource * [std::experimental::fundamentals_v2::pmr::new_delete_resource](#) () noexcept
- memory_resource * [std::experimental::fundamentals_v2::pmr::null_memory_resource](#) () noexcept
- bool [std::experimental::fundamentals_v2::pmr::operator!=](#) (const memory_resource &__a, const memory_resource &__b) noexcept
- template<class _Tp1 , class _Tp2 >
bool [std::experimental::fundamentals_v2::pmr::operator!=](#) (const polymorphic_allocator< _Tp1 > &__a, const polymorphic_allocator< _Tp2 > &__b) noexcept
- bool [std::experimental::fundamentals_v2::pmr::operator==](#) (const memory_resource &__a, const memory_resource &__b) noexcept
- template<class _Tp1 , class _Tp2 >
bool [std::experimental::fundamentals_v2::pmr::operator==](#) (const polymorphic_allocator< _Tp1 > &__a, const polymorphic_allocator< _Tp2 > &__b) noexcept
- memory_resource * [std::experimental::fundamentals_v2::pmr::set_default_resource](#) (memory_resource *__r) noexcept

6.365.1 Detailed Description

This is a TS C++ Library header.

6.366 memoryfwd.h File Reference

Classes

- class [std::allocator](#)< _Tp >
- struct [std::uses_allocator](#)< _Tp, _Alloc >

Namespaces

- [std](#)

6.366.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

6.367 merge.h File Reference

Namespaces

- [__gnu_parallel](#)

Functions

- `template<typename _RAIter1, typename _RAIter2, typename _OutputIterator, typename _DifferenceTp, typename _Compare >
_OutputIterator __gnu_parallel::__merge_advance (_RAIter1 &__begin1, _RAIter1 __end1, _RAIter2 &__begin2, _RAIter2 __end2, _OutputIterator __target, _DifferenceTp __max_length, _Compare __comp)`
- `template<typename _RAIter1, typename _RAIter2, typename _OutputIterator, typename _DifferenceTp, typename _Compare >
_OutputIterator __gnu_parallel::__merge_advance_movc (_RAIter1 &__begin1, _RAIter1 __end1, _RAIter2 &__begin2, _RAIter2 __end2, _OutputIterator __target, _DifferenceTp __max_length, _Compare __comp)`
- `template<typename _RAIter1, typename _RAIter2, typename _OutputIterator, typename _DifferenceTp, typename _Compare >
_OutputIterator __gnu_parallel::__merge_advance_usual (_RAIter1 &__begin1, _RAIter1 __end1, _RAIter2 &__begin2, _RAIter2 __end2, _OutputIterator __target, _DifferenceTp __max_length, _Compare __comp)`
- `template<typename _RAIter1, typename _RAIter2, typename _RAIter3, typename _Compare >
_RAIter3 __gnu_parallel::__parallel_merge_advance (_RAIter1 &__begin1, _RAIter1 __end1, _RAIter2 &__begin2, _RAIter2 __end2, _RAIter3 __target, typename std::iterator_traits<_RAIter1 >::difference_type __max_length, _Compare __comp)`
- `template<typename _RAIter1, typename _RAIter3, typename _Compare >
_RAIter3 __gnu_parallel::__parallel_merge_advance (_RAIter1 &__begin1, _RAIter1 __end1, _RAIter1 &__begin2, _RAIter1 __end2, _RAIter3 __target, typename std::iterator_traits<_RAIter1 >::difference_type __max_length, _Compare __comp)`

6.367.1 Detailed Description

Parallel implementation of `std::merge()`. This file is a GNU parallel extension to the Standard C++ Library.

6.368 `messages_members.h` File Reference

Namespaces

- [std](#)

6.368.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

6.369 `mod_based_range_hashing.hpp` File Reference

Classes

- class [__gnu_pbds::detail::mod_based_range_hashing< Size_Type >](#)

Namespaces

- [__gnu_pbds](#)

6.369.1 Detailed Description

Contains a range hashing policy base.

6.370 move.h File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_FORWARD(_Tp, __val)`
- `#define _GLIBCXX_MOVE(__val)`

Functions

- `template<typename _Tp >`
`_Tp * std::__addressof (_Tp &__r) noexcept`
- `template<typename _Tp, typename _Up = _Tp>`
`_Tp std::__exchange (_Tp &__obj, _Up &&__new_val)`
- `template<typename _Tp >`
`_Tp * std::addressof (_Tp &__r) noexcept`
- `template<typename _Tp >`
`constexpr _Tp && std::forward (typename std::remove_reference< _Tp >::type &__t) noexcept`
- `template<typename _Tp >`
`constexpr _Tp && std::forward (typename std::remove_reference< _Tp >::type &&__t) noexcept`
- `template<typename _Tp >`
`constexpr std::remove_reference< _Tp >::type && std::move (_Tp &&__t) noexcept`
- `template<typename _Tp >`
`constexpr conditional< __move_if_noexcept_cond< _Tp >::value, const _Tp &, _Tp && >::type std::move_if_noexcept (_Tp &__x) noexcept`
- `template<typename _Tp >`
`enable_if< __and< is_move_constructible< _Tp >, is_move_assignable< _Tp > >::value >::type std::swap (_Tp &__a, _Tp &__b) noexcept(__and< is_nothrow_move_constructible< _Tp >, is_nothrow_move_assignable< _Tp > >::value)`
- `template<typename _Tp, size_t _Nm>`
`enable_if< __is_swappable< _Tp >::value >::type std::swap (_Tp(&__a)[_Nm], _Tp(&__b)[_Nm]) noexcept(__is_nothrow_swappable< _Tp >::value)`

6.370.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<utility>`.

6.371 `mt_allocator.h` File Reference

Classes

- struct [__gnu_cxx::__common_pool_policy<_PoolTp, _Thread>](#)
- class [__gnu_cxx::__mt_alloc<_Tp, _Poolp>](#)
- class [__gnu_cxx::__mt_alloc_base<_Tp>](#)
- struct [__gnu_cxx::__per_type_pool_policy<_Tp, _PoolTp, _Thread>](#)
- class [__gnu_cxx::__pool<_Thread>](#)
- class [__gnu_cxx::__pool<false>](#)
- class [__gnu_cxx::__pool<true>](#)
- struct [__gnu_cxx::__pool_base](#)

Namespaces

- [__gnu_cxx](#)

Macros

- `#define __thread_default`

Typedefs

- typedef void(* [__gnu_cxx::__destroy_handler](#)) (void *)

Functions

- template<typename _Tp, typename _Poolp>
bool [__gnu_cxx::operator!=](#) (const [__mt_alloc<_Tp, _Poolp>](#) &, const [__mt_alloc<_Tp, _Poolp>](#) &)
- template<typename _Tp, typename _Poolp>
bool [__gnu_cxx::operator==](#) (const [__mt_alloc<_Tp, _Poolp>](#) &, const [__mt_alloc<_Tp, _Poolp>](#) &)

6.371.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.372 `multimap.h` File Reference

Classes

- class [std::__debug::multimap<_Key, _Tp, _Compare, _Allocator>](#)

Namespaces

- [std](#)
- [std::__debug](#)

Functions

- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >
bool std::__debug::operator!= (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >
bool std::__debug::operator< (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >
bool std::__debug::operator<= (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >
bool std::__debug::operator== (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >
bool std::__debug::operator> (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >
bool std::__debug::operator>= (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap< _Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Allocator >
void std::__debug::swap (multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, multimap< _Key, _Tp, _Compare, _Allocator > &__rhs) noexcept(/*conditional */)`

6.372.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.373 `multimap.h` File Reference

Classes

- class [std::__profile::multimap< _Key, _Tp, _Compare, _Allocator >](#)

Namespaces

- [std](#)
- [std::__profile](#)

Functions

- `template<typename _Key , typename _Tp , typename _Compare , typename _Allocator >`
`bool std::__profile::operator!= (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap<`
`_Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key , typename _Tp , typename _Compare , typename _Allocator >`
`bool std::__profile::operator< (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap<`
`_Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key , typename _Tp , typename _Compare , typename _Allocator >`
`bool std::__profile::operator<= (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap<`
`_Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key , typename _Tp , typename _Compare , typename _Allocator >`
`bool std::__profile::operator== (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap<`
`_Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key , typename _Tp , typename _Compare , typename _Allocator >`
`bool std::__profile::operator> (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap<`
`_Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key , typename _Tp , typename _Compare , typename _Allocator >`
`bool std::__profile::operator>= (const multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, const multimap<`
`_Key, _Tp, _Compare, _Allocator > &__rhs)`
- `template<typename _Key , typename _Tp , typename _Compare , typename _Allocator >`
`void std::__profile::swap (multimap< _Key, _Tp, _Compare, _Allocator > &__lhs, multimap< _Key, _Tp, _Compare, _Allocator > &__rhs) noexcept(/*conditional */)`

6.373.1 Detailed Description

This file is a GNU profile extension to the Standard C++ Library.

6.374 multiseq_selection.h File Reference

Classes

- class [__gnu_parallel::_Lexicographic< _T1, _T2, _Compare >](#)
- class [__gnu_parallel::_LexicographicReverse< _T1, _T2, _Compare >](#)

Namespaces

- [__gnu_parallel](#)

Macros

- `#define __S(__i)`
- `#define __S(__i)`

Functions

- `template<typename _RanSeqs, typename _RankType, typename _RankIterator, typename _Compare >`
`void __gnu_parallel::multiseq_partition (_RanSeqs __begin_seqs, _RanSeqs __end_seqs, _RankType __rank,`
`_RankIterator __begin_offsets, _Compare __comp=std::less< typename std::iterator_traits< typename std::iterator_traits< _RanSeqs >::value_type::first_type >::value_type >>())`
- `template<typename _Tp, typename _RanSeqs, typename _RankType, typename _Compare >`
`_Tp __gnu_parallel::multiseq_selection (_RanSeqs __begin_seqs, _RanSeqs __end_seqs, _RankType __rank,`
`_RankType &__offset, _Compare __comp=std::less< _Tp >())`

6.374.1 Detailed Description

Functions to find elements of a certain global `__rank` in multiple sorted sequences. Also serves for splitting such sequence sets.

The algorithm description can be found in

P. J. Varman, S. D. Scheufler, B. R. Iyer, and G. R. Ricard. Merging Multiple Lists on Hierarchical-Memory Multiprocessors. *Journal of Parallel and Distributed Computing*, 12(2):171–177, 1991.

This file is a GNU parallel extension to the Standard C++ Library.

6.375 multiset.h File Reference

Classes

- class `std::__debug::multiset< _Key, _Compare, _Allocator >`

Namespaces

- `std`
- `std::__debug`

Functions

- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool std::__debug::operator!= (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key,`
`_Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool std::__debug::operator< (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key,`
`_Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool std::__debug::operator<= (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key,`
`_Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool std::__debug::operator== (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key,`
`_Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool std::__debug::operator> (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key,`
`_Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool std::__debug::operator>= (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key,`
`_Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`void std::__debug::swap (multiset< _Key, _Compare, _Allocator > &__x, multiset< _Key, _Compare, _Allocator`
`> &__y) noexcept(/*conditional */)`

6.375.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.376 multiset.h File Reference

Classes

- class [std::__profile::multiset<_Key, _Compare, _Allocator >](#)

Namespaces

- [std](#)
- [std::__profile](#)

Functions

- `template<typename _Key, typename _Compare, typename _Allocator >
bool std::__profile::operator!= (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >
bool std::__profile::operator< (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >
bool std::__profile::operator<= (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >
bool std::__profile::operator== (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >
bool std::__profile::operator> (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >
bool std::__profile::operator>= (const multiset< _Key, _Compare, _Allocator > &__lhs, const multiset< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >
void std::__profile::swap (multiset< _Key, _Compare, _Allocator > &__x, multiset< _Key, _Compare, _Allocator > &__y) noexcept(/*conditional */)`

6.376.1 Detailed Description

This file is a GNU profile extension to the Standard C++ Library.

6.377 multiway_merge.h File Reference

Classes

- struct [__gnu_parallel::__multiway_merge_3_variant_sentinel_switch](#)< __sentinels, _RAIterlterator, _RAIter3, [_DifferenceTp](#), [_Compare](#) >
- struct [__gnu_parallel::__multiway_merge_3_variant_sentinel_switch](#)< true, _RAIterlterator, _RAIter3, [_DifferenceTp](#), [_Compare](#) >
- struct [__gnu_parallel::__multiway_merge_4_variant_sentinel_switch](#)< __sentinels, _RAIterlterator, _RAIter3, [_DifferenceTp](#), [_Compare](#) >
- struct [__gnu_parallel::__multiway_merge_4_variant_sentinel_switch](#)< true, _RAIterlterator, _RAIter3, [_DifferenceTp](#), [_Compare](#) >
- struct [__gnu_parallel::__multiway_merge_k_variant_sentinel_switch](#)< __sentinels, __stable, _RAIterlterator, [_RAIter3](#), [_DifferenceTp](#), [_Compare](#) >
- struct [__gnu_parallel::__multiway_merge_k_variant_sentinel_switch](#)< false, __stable, _RAIterlterator, _RAIter3, [_DifferenceTp](#), [_Compare](#) >
- class [__gnu_parallel::_Guardedlterator](#)< _RAIter, [_Compare](#) >
- struct [__gnu_parallel::_LoserTreeTraits](#)< [_Tp](#) >
- struct [__gnu_parallel::_SamplingSorter](#)< __stable, _RAIter, [_StrictWeakOrdering](#) >
- struct [__gnu_parallel::_SamplingSorter](#)< false, _RAIter, [_StrictWeakOrdering](#) >

Namespaces

- [__gnu_parallel](#)

Macros

- [#define GLIBCXX_PARALLEL_DECISION](#)(__a, __b, __c, __d)
- [#define GLIBCXX_PARALLEL_LENGTH](#)(__s)
- [#define GLIBCXX_PARALLEL_MERGE_3_CASE](#)(__a, __b, __c, __c0, __c1)
- [#define GLIBCXX_PARALLEL_MERGE_4_CASE](#)(__a, __b, __c, __d, __c0, __c1, __c2)

Functions

- [template<typename _RAIter1, typename _RAIter2, typename _Outputlterator, typename _DifferenceTp, typename _Compare> _Outputlterator __gnu_parallel::merge_advance](#) (_RAIter1 & __begin1, _RAIter1 __end1, _RAIter2 & __begin2, _RAIter2 __end2, _Outputlterator __target, [_DifferenceTp](#) __max_length, [_Compare](#) __comp)
- [template<bool __stable, bool __sentinels, typename _RAIterlterator, typename _RAIter3, typename _DifferenceTp, typename _Compare> _RAIter3 __gnu_parallel::sequential_multiway_merge](#) (_RAIterlterator __seqs_begin, _RAIterlterator __seqs_end, _RAIter3 __target, const typename std::iterator_traits< typename std::iterator_traits< _RAIterlterator >::value_type::first_type >::value_type & __sentinel, [_DifferenceTp](#) __length, [_Compare](#) __comp)
- [template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare> _RAIterOut __gnu_parallel::multiway_merge](#) (_RAIterPairlterator __seqs_begin, _RAIterPairlterator __seqs_end, _RAIterOut __target, [_DifferenceTp](#) __length, [_Compare](#) __comp, [__gnu_parallel::sequential_tag](#))
- [template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare> _RAIterOut __gnu_parallel::multiway_merge](#) (_RAIterPairlterator __seqs_begin, _RAIterPairlterator __seqs_end, _RAIterOut __target, [_DifferenceTp](#) __length, [_Compare](#) __comp, [__gnu_parallel::exact_tag](#) __tag)

- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut gnu_parallel::multiway_merge (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, gnu_parallel::sampling_tag __tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut gnu_parallel::multiway_merge (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, parallel_tag __tag=parallel_tag(0))`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut gnu_parallel::multiway_merge (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, default_parallel_tag __tag)`
- `template<template< typename RAI, typename C > class iterator, typename _RAIterIterator, typename _RAIter3, typename _DifferenceTp, typename _Compare >`
`_RAIter3 gnu_parallel::multiway_merge_3_variant (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 __target, _DifferenceTp __length, _Compare __comp)`
- `template<template< typename RAI, typename C > class iterator, typename _RAIterIterator, typename _RAIter3, typename _DifferenceTp, typename _Compare >`
`_RAIter3 gnu_parallel::multiway_merge_4_variant (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 __target, _DifferenceTp __length, _Compare __comp)`
- `template<bool __stable, typename _RAIterIterator, typename _Compare, typename _DifferenceType >`
`void gnu_parallel::multiway_merge_exact_splitting (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _DifferenceType __length, _DifferenceType __total_length, _Compare __comp, std::vector< std::pair< _DifferenceType, _DifferenceType > > * __pieces)`
- `template<typename _LT, typename _RAIterIterator, typename _RAIter3, typename _DifferenceTp, typename _Compare >`
`_RAIter3 gnu_parallel::multiway_merge_loser_tree (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 __target, _DifferenceTp __length, _Compare __comp)`
- `template<typename UnguardedLoserTree, typename _RAIterIterator, typename _RAIter3, typename _DifferenceTp, typename _Compare >`
`_RAIter3 gnu_parallel::multiway_merge_loser_tree_sentinel (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 __target, const typename std::iterator_traits< typename std::iterator_traits< _RAIterIterator >::value_type::first_type >::value_type & __sentinel, _DifferenceTp __length, _Compare __comp)`
- `template<typename _LT, typename _RAIterIterator, typename _RAIter3, typename _DifferenceTp, typename _Compare >`
`_RAIter3 gnu_parallel::multiway_merge_loser_tree_unguarded (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _RAIter3 __target, const typename std::iterator_traits< typename std::iterator_traits< _RAIterIterator >::value_type::first_type >::value_type & __sentinel, _DifferenceTp __length, _Compare __comp)`
- `template<bool __stable, typename _RAIterIterator, typename _Compare, typename _DifferenceType >`
`void gnu_parallel::multiway_merge_sampling_splitting (_RAIterIterator __seqs_begin, _RAIterIterator __seqs_end, _DifferenceType __length, _DifferenceType __total_length, _Compare __comp, std::vector< std::pair< _DifferenceType, _DifferenceType > > * __pieces)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut gnu_parallel::multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, gnu_parallel::sequential_tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut gnu_parallel::multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, gnu_parallel::exact_tag __tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut gnu_parallel::multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, sampling_tag __tag)`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut gnu_parallel::multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, parallel_tag __tag=parallel_tag(0))`
- `template<typename _RAIterPairIterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut gnu_parallel::multiway_merge_sentinels (_RAIterPairIterator __seqs_begin, _RAIterPairIterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, default_parallel_tag __tag)`

- `template<bool __stable, bool __sentinels, typename _RAIterlterator, typename _RAIter3, typename _DifferenceTp, typename _Splitter, typename _Compare >`
`_RAIter3 __gnu_parallel::parallel_multiway_merge (_RAIterlterator __seqs_begin, _RAIterlterator __seqs_end,`
`_RAIter3 __target, _Splitter __splitter, _DifferenceTp __length, _Compare __comp, _ThreadIndex __num ↵`
`threads)`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut __gnu_parallel::stable_multiway_merge (_RAIterPairlterator __seqs_begin, _RAIterPairlterator ↵`
`__seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, __gnu_parallel::sequential_tag)`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut __gnu_parallel::stable_multiway_merge (_RAIterPairlterator __seqs_begin, _RAIterPairlterator ↵`
`__seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, __gnu_parallel::exact_tag __tag)`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut __gnu_parallel::stable_multiway_merge (_RAIterPairlterator __seqs_begin, _RAIterPairlterator ↵`
`__seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, sampling_tag __tag)`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut __gnu_parallel::stable_multiway_merge (_RAIterPairlterator __seqs_begin, _RAIterPairlterator ↵`
`__seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, parallel_tag __tag=parallel ↵`
`tag(0))`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut __gnu_parallel::stable_multiway_merge (_RAIterPairlterator __seqs_begin, _RAIterPairlterator ↵`
`__seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, default_parallel_tag __tag)`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut __gnu_parallel::stable_multiway_merge_sentinels (_RAIterPairlterator __seqs_begin, _RAIter ↵`
`Pairlterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, __gnu_parallel ↵`
`::sequential_tag)`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut __gnu_parallel::stable_multiway_merge_sentinels (_RAIterPairlterator __seqs_begin, _RAIter ↵`
`Pairlterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, __gnu_parallel ↵`
`::exact_tag __tag)`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut __gnu_parallel::stable_multiway_merge_sentinels (_RAIterPairlterator __seqs_begin, _RAIter ↵`
`Pairlterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, sampling_tag __tag)`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut __gnu_parallel::stable_multiway_merge_sentinels (_RAIterPairlterator __seqs_begin, _RAIter ↵`
`Pairlterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, parallel_tag ↵`
`__tag=parallel_tag(0))`
- `template<typename _RAIterPairlterator, typename _RAIterOut, typename _DifferenceTp, typename _Compare >`
`_RAIterOut __gnu_parallel::stable_multiway_merge_sentinels (_RAIterPairlterator __seqs_begin, _RAIter ↵`
`Pairlterator __seqs_end, _RAIterOut __target, _DifferenceTp __length, _Compare __comp, default_parallel_tag`
`__tag)`

6.377.1 Detailed Description

Implementation of sequential and parallel multiway merge.

Explanations on the high-speed merging routines in the appendix of

P. Sanders. Fast priority queues for cached memory. ACM Journal of Experimental Algorithmics, 5, 2000.

This file is a GNU parallel extension to the Standard C++ Library.

6.377.2 Macro Definition Documentation

6.377.2.1 `#define _GLIBCXX_PARALLEL_LENGTH(__s)`

Length of a sequence described by a pair of iterators.

Definition at line 54 of file `multiway_merge.h`.

Referenced by `__gnu_parallel::__sequential_multiway_merge()`, `__gnu_parallel::multiway_merge_exact_splitting()`, `__gnu_parallel::multiway_merge_loser_tree()`, `__gnu_parallel::multiway_merge_sampling_splitting()`, and `__gnu_parallel::parallel_multiway_merge()`.

6.378 `multiway_mergesort.h` File Reference

Classes

- struct [__gnu_parallel::Piece<_DifferenceTp>](#)
- struct [__gnu_parallel::PMWMSortingData<_RAIter>](#)
- struct [__gnu_parallel::SplitConsistently<__exact, _RAIter, _Compare, _SortingPlacesIterator>](#)
- struct [__gnu_parallel::SplitConsistently<false, _RAIter, _Compare, _SortingPlacesIterator>](#)
- struct [__gnu_parallel::SplitConsistently<true, _RAIter, _Compare, _SortingPlacesIterator>](#)

Namespaces

- [__gnu_parallel](#)

Functions

- template<typename _RAIter, typename _DifferenceTp>
void [__gnu_parallel::__determine_samples](#) (_PMWMSortingData<_RAIter> * __sd, _DifferenceTp __num←_samples)
- template<bool __stable, bool __exact, typename _RAIter, typename _Compare>
void [__gnu_parallel::parallel_sort_mwms](#) (_RAIter __begin, _RAIter __end, _Compare __comp, _ThreadIndex __num_threads)
- template<bool __stable, bool __exact, typename _RAIter, typename _Compare>
void [__gnu_parallel::parallel_sort_mwms_pu](#) (_PMWMSortingData<_RAIter> * __sd, _Compare & __comp)

6.378.1 Detailed Description

Parallel multiway merge sort. This file is a GNU parallel extension to the Standard C++ Library.

6.379 `mutex` File Reference

Classes

- struct [std::once_flag](#)
- class [std::recursive_mutex](#)
- class [std::recursive_timed_mutex](#)
- class [std::timed_mutex](#)

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_MUTEX`
- `__thread void * std::__once_callable`
- `__thread void(* std::__once_call)()`
- `template<typename _Lock >
unique_lock< _Lock > std::__try_to_lock (_Lock &__l)`
- `template<typename _Lock1 , typename _Lock2 , typename... _Lock3>
int std::try_lock (_Lock1 &__l1, _Lock2 &__l2, _Lock3 &...__l3)`
- `template<typename _L1 , typename _L2 , typename... _L3>
void std::lock (_L1 &__l1, _L2 &__l2, _L3 &...__l3)`
- `void std::__once_proxy (void)`
- `template<typename _Callable , typename... _Args>
void std::call_once (once_flag &__once, _Callable &&__f, _Args &&...__args)`

6.379.1 Detailed Description

This is a Standard C++ Library header.

6.380 nested_exception.h File Reference

Classes

- class [std::nested_exception](#)

Namespaces

- [std](#)

Typedefs

- `template<typename _Tp >
using std::__rethrow_if_nested_cond = typename enable_if< __and< is_polymorphic< _Tp >, __or< __↵
not< is_base_of< nested_exception, _Tp >>, is_convertible< _Tp *, nested_exception * >>>::value >::type`

Functions

- `template<typename _Ex >`
`__rethrow_if_nested_cond< _Ex > std::__rethrow_if_nested_impl (const _Ex * __ptr)`
- `void std::__rethrow_if_nested_impl (const void *)`
- `template<typename _Tp >`
`void std::__throw_with_nested_impl (_Tp && __t, true_type)`
- `template<typename _Tp >`
`void std::__throw_with_nested_impl (_Tp && __t, false_type)`
- `template<typename _Ex >`
`void std::rethrow_if_nested (const _Ex & __ex)`
- `template<typename _Tp >`
`void std::throw_with_nested (_Tp && __t)`

6.380.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<exception>`.

6.381 new File Reference

Classes

- class `std::bad_alloc`

Namespaces

- `std`

Typedefs

- `typedef void(* std::new_handler) ()`

Functions

- `new_handler std::get_new_handler () noexcept`
- `new_handler std::set_new_handler (new_handler) throw ()`
- `void * operator new (std::size_t) __attribute__((__externally_visible__))`
- `void * operator new[] (std::size_t) __attribute__((__externally_visible__))`
- `void operator delete (void *) noexcept __attribute__((__externally_visible__))`
- `void operator delete[] (void *) noexcept __attribute__((__externally_visible__))`
- `void * operator new (std::size_t, const std::nothrow_t &) noexcept __attribute__((__externally_visible__))`
- `void * operator new[] (std::size_t, const std::nothrow_t &) noexcept __attribute__((__externally_visible__))`
- `void operator delete (void *, const std::nothrow_t &) noexcept __attribute__((__externally_visible__))`
- `void operator delete[] (void *, const std::nothrow_t &) noexcept __attribute__((__externally_visible__))`
- `void * operator new (std::size_t, void * __p) noexcept`
- `void * operator new[] (std::size_t, void * __p) noexcept`
- `void operator delete (void *, void *) noexcept`
- `void operator delete[] (void *, void *) noexcept`

Variables

- `const nothrow_t` **`std::nothrow`**

6.381.1 Detailed Description

This is a Standard C++ Library header.

The header `new` defines several functions to manage dynamic memory and handling memory allocation errors; see http://gcc.gnu.org/onlinedocs/libstdc++/18_support/howto.html#4 for more.

6.381.2 Function Documentation

6.381.2.1 `void operator delete (void *) [nothrow]`

These are replaceable signatures:

- normal single new and delete (no arguments, throw `bad_alloc` on error)
- normal array new and delete (same)
- `nothrow` single new and delete (take a `nothrow` argument, return `NULL` on error)
- `nothrow` array new and delete (same)

Placement new and delete signatures (take a memory address argument, does nothing) may not be replaced by a user's program.

6.381.2.2 `void operator delete (void *, const std::nothrow_t &) [nothrow]`

These are replaceable signatures:

- normal single new and delete (no arguments, throw `bad_alloc` on error)
- normal array new and delete (same)
- `nothrow` single new and delete (take a `nothrow` argument, return `NULL` on error)
- `nothrow` array new and delete (same)

Placement new and delete signatures (take a memory address argument, does nothing) may not be replaced by a user's program.

6.381.2.3 `void operator delete (void *, void *) [inline], [noexcept]`

These are replaceable signatures:

- normal single new and delete (no arguments, throw `bad_alloc` on error)
- normal array new and delete (same)
- `nothrow` single new and delete (take a `nothrow` argument, return `NULL` on error)
- `nothrow` array new and delete (same)

Placement new and delete signatures (take a memory address argument, does nothing) may not be replaced by a user's program.

Definition at line 152 of file `new`.

6.381.2.4 `void operator delete[] (void *) [noexcept]`

These are replaceable signatures:

- normal single new and delete (no arguments, throw `bad_alloc` on error)
- normal array new and delete (same)
- `nothrow` single new and delete (take a `nothrow` argument, return `NULL` on error)
- `nothrow` array new and delete (same)

Placement new and delete signatures (take a memory address argument, does nothing) may not be replaced by a user's program.

6.381.2.5 `void operator delete[] (void *, const std::nothrow_t &) [noexcept]`

These are replaceable signatures:

- normal single new and delete (no arguments, throw `bad_alloc` on error)
- normal array new and delete (same)
- `nothrow` single new and delete (take a `nothrow` argument, return `NULL` on error)
- `nothrow` array new and delete (same)

Placement new and delete signatures (take a memory address argument, does nothing) may not be replaced by a user's program.

6.381.2.6 `void operator delete[] (void *, void *) [inline], [nothrow]`

These are replaceable signatures:

- normal single new and delete (no arguments, throw `bad_alloc` on error)
- normal array new and delete (same)
- `nothrow` single new and delete (take a `nothrow` argument, return `NULL` on error)
- `nothrow` array new and delete (same)

Placement new and delete signatures (take a memory address argument, does nothing) may not be replaced by a user's program.

Definition at line 153 of file `new`.

6.381.2.7 `void* operator new (std::size_t)`

These are replaceable signatures:

- normal single new and delete (no arguments, throw `bad_alloc` on error)
- normal array new and delete (same)
- `nothrow` single new and delete (take a `nothrow` argument, return `NULL` on error)
- `nothrow` array new and delete (same)

Placement new and delete signatures (take a memory address argument, does nothing) may not be replaced by a user's program.

6.381.2.8 `void* operator new (std::size_t, const std::nothrow_t &) [nothrow]`

These are replaceable signatures:

- normal single new and delete (no arguments, throw `bad_alloc` on error)
- normal array new and delete (same)
- `nothrow` single new and delete (take a `nothrow` argument, return `NULL` on error)
- `nothrow` array new and delete (same)

Placement new and delete signatures (take a memory address argument, does nothing) may not be replaced by a user's program.

6.381.2.9 `void* operator new (std::size_t, void * __p) [inline], [noexcept]`

These are replaceable signatures:

- normal single new and delete (no arguments, throw `bad_alloc` on error)
- normal array new and delete (same)
- `nothrow` single new and delete (take a `nothrow` argument, return `NULL` on error)
- `nothrow` array new and delete (same)

Placement new and delete signatures (take a memory address argument, does nothing) may not be replaced by a user's program.

Definition at line 146 of file `new`.

6.381.2.10 `void* operator new[] (std::size_t)`

These are replaceable signatures:

- normal single new and delete (no arguments, throw `bad_alloc` on error)
- normal array new and delete (same)
- `nothrow` single new and delete (take a `nothrow` argument, return `NULL` on error)
- `nothrow` array new and delete (same)

Placement new and delete signatures (take a memory address argument, does nothing) may not be replaced by a user's program.

6.381.2.11 `void* operator new[] (std::size_t, const std::nothrow_t &) [noexcept]`

These are replaceable signatures:

- normal single new and delete (no arguments, throw `bad_alloc` on error)
- normal array new and delete (same)
- `nothrow` single new and delete (take a `nothrow` argument, return `NULL` on error)
- `nothrow` array new and delete (same)

Placement new and delete signatures (take a memory address argument, does nothing) may not be replaced by a user's program.

6.381.2.12 `void* operator new[](std::size_t, void *__p) [inline], [noexcept]`

These are replaceable signatures:

- normal single new and delete (no arguments, throw `bad_alloc` on error)
- normal array new and delete (same)
- `nothrow` single new and delete (take a `nothrow` argument, return `NULL` on error)
- `nothrow` array new and delete (same)

Placement new and delete signatures (take a memory address argument, does nothing) may not be replaced by a user's program.

Definition at line 148 of file `new`.

6.382 new_allocator.h File Reference

Classes

- class [__gnu_cxx::new_allocator< _Tp >](#)

Namespaces

- [__gnu_cxx](#)

Functions

- `template<typename _Tp >`
`bool __gnu_cxx::operator!= (const new_allocator< _Tp > &, const new_allocator< _Tp > &)`
- `template<typename _Tp >`
`bool __gnu_cxx::operator== (const new_allocator< _Tp > &, const new_allocator< _Tp > &)`

6.382.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.383 node.hpp File Reference

Classes

- struct [__gnu_pbds::detail::left_child_next_sibling_heap_node_< _Value, _Metadata, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

6.383.1 Detailed Description

Contains an implementation struct for this type of heap's node.

6.384 node.hpp File Reference

Classes

- struct [__gnu_pbds::detail::rb_tree_node_< Value_Type, Metadata, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

6.384.1 Detailed Description

Contains an implementation for rb_tree_.

6.385 node.hpp File Reference

Classes

- struct [__gnu_pbds::detail::splay_tree_node_< Value_Type, Metadata, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

6.385.1 Detailed Description

Contains an implementation struct for splay_tree_'s node.

6.386 node_iterators.hpp File Reference

Classes

- class [__gnu_pbds::detail::bin_search_tree_const_node_it_< Node, Const_Iterator, Iterator, _Alloc >](#)
- class [__gnu_pbds::detail::bin_search_tree_node_it_< Node, Const_Iterator, Iterator, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_TREE_CONST_NODE_ITERATOR_CLASS_C_DEC`
- `#define PB_DS_TREE_NODE_ITERATOR_CLASS_C_DEC`

6.386.1 Detailed Description

Contains an implementation class for `bin_search_tree_`.

6.387 node_iterators.hpp File Reference

Classes

- class [__gnu_pbds::detail::ov_tree_node_const_it_< Value_Type, Metadata_Type, _Alloc >](#)
- class [__gnu_pbds::detail::ov_tree_node_it_< Value_Type, Metadata_Type, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_OV_TREE_CONST_NODE_ITERATOR_C_DEC`
- `#define PB_DS_OV_TREE_NODE_ITERATOR_C_DEC`

6.387.1 Detailed Description

Contains an implementation class for `ov_tree_`.

6.388 node_metadata_selector.hpp File Reference

Classes

- struct [__gnu_pbds::detail::tree_metadata_helper< Node_Update, _BTp >](#)
- struct [__gnu_pbds::detail::tree_metadata_helper< Node_Update, false >](#)
- struct [__gnu_pbds::detail::tree_metadata_helper< Node_Update, true >](#)
- struct [__gnu_pbds::detail::tree_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

6.388.1 Detailed Description

Contains an implementation class for trees.

6.389 node_metadata_selector.hpp File Reference

Classes

- [struct __gnu_pbds::detail::trie_metadata_helper< Node_Update, _BTp >](#)
- [struct __gnu_pbds::detail::trie_metadata_helper< Node_Update, false >](#)
- [struct __gnu_pbds::detail::trie_metadata_helper< Node_Update, true >](#)
- [struct __gnu_pbds::detail::trie_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

6.389.1 Detailed Description

Contains an implementation class for tries.

6.390 null_node_metadata.hpp File Reference

Classes

- [struct __gnu_pbds::detail::dumnode_const_iterator< Key, Data, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

6.390.1 Detailed Description

Contains an implementation class for tree-like classes.

6.391 numeric File Reference

Macros

- `#define _GLIBCXX_NUMERIC`

6.391.1 Detailed Description

This is a Standard C++ Library header.

6.392 numeric File Reference

Namespaces

- [__gnu_cxx](#)

Macros

- `#define _EXT_NUMERIC`

Functions

- `template<typename _Tp, typename _Integer, typename _MonoidOperation >
_Tp __gnu_cxx::__power (_Tp __x, _Integer __n, _MonoidOperation __monoid_op)`
- `template<typename _Tp, typename _Integer >
_Tp __gnu_cxx::__power (_Tp __x, _Integer __n)`
- `template<typename _Tp, typename _Integer, typename _MonoidOperation >
_Tp __gnu_cxx::power (_Tp __x, _Integer __n, _MonoidOperation __monoid_op)`
- `template<typename _Tp, typename _Integer >
_Tp __gnu_cxx::power (_Tp __x, _Integer __n)`

6.392.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGL STL subset).

6.393 numeric File Reference

Namespaces

- [std](#)
- [std::__parallel](#)

Macros

- `#define _GLIBCXX_PARALLEL_NUMERIC_H`

Functions

- `template<typename _Iter, typename _Tp, typename _IteratorTag >`
`_Tp std::__parallel::__accumulate_switch (_Iter __begin, _Iter __end, _Tp __init, _IteratorTag)`
- `template<typename _Iter, typename _Tp, typename _BinaryOperation, typename _IteratorTag >`
`_Tp std::__parallel::__accumulate_switch (_Iter __begin, _Iter __end, _Tp __init, _BinaryOperation __↵`
`binary_op, _IteratorTag)`
- `template<typename __RAIter, typename _Tp, typename _BinaryOperation >`
`_Tp std::__parallel::__accumulate_switch (__RAIter __begin, __RAIter __end, _Tp __init, _BinaryOperation`
`__binary_op, random_access_iterator_tag, __gnu_parallel::__Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation, typename _IteratorTag1, typename _IteratorTag2 >`
`_OutputIterator std::__parallel::__adjacent_difference_switch (_Iter __begin, _Iter __end, _OutputIterator ↵`
`__result, _BinaryOperation __bin_op, _IteratorTag1, _IteratorTag2)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::__parallel::__adjacent_difference_switch (_Iter __begin, _Iter __end, _OutputIterator`
`__result, _BinaryOperation __bin_op, random_access_iterator_tag, random_access_iterator_tag, __gnu_↵`
`parallel::__Parallelism __parallelism_tag)`
- `template<typename _RAIter1, typename _RAIter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 >`
`_Tp std::__parallel::__inner_product_switch (_RAIter1 __first1, _RAIter1 __last1, _RAIter2 __first2, _Tp ↵`
`init, _BinaryFunction1 __binary_op1, _BinaryFunction2 __binary_op2, random_access_iterator_tag, random_↵`
`access_iterator_tag, __gnu_parallel::__Parallelism __parallelism_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2, typename ↵`
`IteratorTag1, typename _IteratorTag2 >`
`_Tp std::__parallel::__inner_product_switch (_Iter1 __first1, _Iter1 __last1, _Iter2 __first2, _Tp __init, ↵`
`BinaryFunction1 __binary_op1, _BinaryFunction2 __binary_op2, _IteratorTag1, _IteratorTag2)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation, typename _IteratorTag1, typename _IteratorTag2 >`
`_OutputIterator std::__parallel::__partial_sum_switch (_Iter __begin, _Iter __end, _OutputIterator __result,`
`_BinaryOperation __bin_op, _IteratorTag1, _IteratorTag2)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::__parallel::__partial_sum_switch (_Iter __begin, _Iter __end, _OutputIterator __result,`
`_BinaryOperation __bin_op, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter, typename _Tp >`
`_Tp std::__parallel::__accumulate (_Iter, _Iter, _Tp)`
- `template<typename _Iter, typename _Tp >`
`_Tp std::__parallel::__accumulate (_Iter, _Iter, _Tp, __gnu_parallel::__sequential_tag)`
- `template<typename _Iter, typename _Tp >`
`_Tp std::__parallel::__accumulate (_Iter, _Iter, _Tp, __gnu_parallel::__Parallelism)`
- `template<typename _Iter, typename _Tp, typename _BinaryOperation >`
`_Tp std::__parallel::__accumulate (_Iter __begin, _Iter __end, _Tp __init, _BinaryOperation __binary_op, ↵`
`gnu_parallel::__sequential_tag)`
- `template<typename _Iter, typename _Tp, typename _BinaryOperation >`
`_Tp std::__parallel::__accumulate (_Iter __begin, _Iter __end, _Tp __init, _BinaryOperation __binary_op, ↵`
`gnu_parallel::__Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _Tp, typename _BinaryOperation >`
`_Tp std::__parallel::__accumulate (_Iter __begin, _Iter __end, _Tp __init, _BinaryOperation __binary_op)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator std::__parallel::__adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result, ↵`
`gnu_parallel::__sequential_tag)`

- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::parallel::adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result, _↵`
`BinaryOperation __bin_op, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator std::parallel::adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result, _↵`
`__gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator std::parallel::adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::parallel::adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result, _↵`
`BinaryOperation __binary_op, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::parallel::adjacent_difference (_Iter __begin, _Iter __end, _OutputIterator __result, _↵`
`BinaryOperation __binary_op)`
- `template<typename _Iter1, typename _Iter2, typename _Tp >`
`_Tp std::parallel::inner_product (_Iter1, _Iter1, _Iter2, _Tp)`
- `template<typename _Iter1, typename _Iter2, typename _Tp >`
`_Tp std::parallel::inner_product (_Iter1, _Iter1, _Iter2, _Tp, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Tp >`
`_Tp std::parallel::inner_product (_Iter1, _Iter1, _Iter2, _Tp, __gnu_parallel::Parallelism)`
- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 >`
`_Tp std::parallel::inner_product (_Iter1, _Iter1, _Iter2, _Tp, _BinaryFunction1, _BinaryFunction2)`
- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 >`
`_Tp std::parallel::inner_product (_Iter1, _Iter1, _Iter2, _Tp, _BinaryFunction1, _BinaryFunction2, __gnu↵`
`__parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 >`
`_Tp std::parallel::inner_product (_Iter1 __first1, _Iter1 __last1, _Iter2 __first2, _Tp __init, _Binary↵`
`Function1 __binary_op1, _BinaryFunction2 __binary_op2, __gnu_parallel::Parallelism __parallelism_tag)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator std::parallel::partial_sum (_Iter __begin, _Iter __end, _OutputIterator __result, __gnu↵`
`__parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::parallel::partial_sum (_Iter __begin, _Iter __end, _OutputIterator __result, _Binary↵`
`Operation __bin_op, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator std::parallel::partial_sum (_Iter __begin, _Iter __end, _OutputIterator __result)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::parallel::partial_sum (_Iter __begin, _Iter __end, _OutputIterator __result, _Binary↵`
`Operation __binary_op)`

6.393.1 Detailed Description

Parallel STL function calls corresponding to `stl_numeric.h`. The functions defined here mainly do case switches and call the actual parallelized versions in other files. Inlining policy: Functions that basically only contain one function call, are declared inline. This file is a GNU parallel extension to the Standard C++ Library.

6.394 numeric File Reference

Namespaces

- [std](#)

Macros

- `#define __cpp_lib_experimental_gcd_lcm`
- `#define _GLIBCXX_EXPERIMENTAL_NUMERIC`

Functions

- `template<typename _Tp >`
`constexpr enable_if_t< __and< is_integral< _Tp >, is_signed< _Tp > >::value, _Tp > std::experimental::fundamentals_v2::__abs (_Tp __val)`
- `template<typename _Tp >`
`constexpr enable_if_t< __and< is_integral< _Tp >, is_unsigned< _Tp > >::value, _Tp > std::experimental::fundamentals_v2::__abs (_Tp __val)`
- `template<typename _Mn, typename _Nn >`
`constexpr common_type_t< _Mn, _Nn > std::experimental::fundamentals_v2::gcd (_Mn __m, _Nn __n)`
- `template<typename _Mn, typename _Nn >`
`constexpr common_type_t< _Mn, _Nn > std::experimental::fundamentals_v2::lcm (_Mn __m, _Nn __n)`

6.394.1 Detailed Description

This is a TS C++ Library header.

6.395 numeric_traits.h File Reference

Namespaces

- [`__gnu_cxx`](#)

Macros

- `#define __glibcxx_digits(_Tp)`
- `#define __glibcxx_digits10(_Tp)`
- `#define __glibcxx_floating(_Tp, _Fval, _Dval, _LDval)`
- `#define __glibcxx_max(_Tp)`
- `#define __glibcxx_max_digits10(_Tp)`
- `#define __glibcxx_max_exponent10(_Tp)`
- `#define __glibcxx_min(_Tp)`
- `#define __glibcxx_signed(_Tp)`

6.395.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.396 numericfwd.h File Reference

Namespaces

- [std](#)
- [std::__parallel](#)

Functions

- `template<typename _Iter, typename _Tp, typename _Tag >
_Tp std::__parallel::__accumulate_switch (_Iter, _Iter, _Tp, _Tag)`
- `template<typename _Iter, typename _Tp, typename _BinaryOper, typename _Tag >
_Tp std::__parallel::__accumulate_switch (_Iter, _Iter, _Tp, _BinaryOper, _Tag)`
- `template<typename _RAIter, typename _Tp, typename _BinaryOper >
_Tp std::__parallel::__accumulate_switch (_RAIter, _RAIter, _Tp, _BinaryOper, random_access_iterator_tag, __gnu_parallel::__Parallelism __parallelism=__gnu_parallel::parallel_unbalanced)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper, typename _Tag1, typename _Tag2 >
_OIter std::__parallel::__adjacent_difference_switch (_Iter, _Iter, _OIter, _BinaryOper, _Tag1, _Tag2)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >
_OIter std::__parallel::__adjacent_difference_switch (_Iter, _Iter, _OIter, _BinaryOper, random_access_iterator_tag, random_access_iterator_tag, __gnu_parallel::__Parallelism __parallelism=__gnu_parallel::parallel_unbalanced)`
- `template<typename _RAIter1, typename _RAIter2, typename _Tp, typename BinaryFunction1, typename BinaryFunction2 >
_Tp std::__parallel::__inner_product_switch (_RAIter1, _RAIter1, _RAIter2, _Tp, BinaryFunction1, BinaryFunction2, random_access_iterator_tag, random_access_iterator_tag, __gnu_parallel::__Parallelism=__gnu_parallel::parallel_unbalanced)`
- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2, typename _Tag1, typename _Tag2 >
_Tp std::__parallel::__inner_product_switch (_Iter1, _Iter1, _Iter2, _Tp, _BinaryFunction1, _BinaryFunction2, _Tag1, _Tag2)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper, typename _Tag1, typename _Tag2 >
_OIter std::__parallel::__partial_sum_switch (_Iter, _Iter, _OIter, _BinaryOper, _Tag1, _Tag2)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >
_OIter std::__parallel::__partial_sum_switch (_Iter, _Iter, _OIter, _BinaryOper, random_access_iterator_tag, random_access_iterator_tag)`
- `template<typename _Iter, typename _Tp >
_Tp std::__parallel::accumulate (_Iter, _Iter, _Tp)`
- `template<typename _Iter, typename _Tp >
_Tp std::__parallel::accumulate (_Iter, _Iter, _Tp, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Tp >
_Tp std::__parallel::accumulate (_Iter, _Iter, _Tp, __gnu_parallel::__Parallelism)`
- `template<typename _Iter, typename _Tp, typename _BinaryOper >
_Tp std::__parallel::accumulate (_Iter, _Iter, _Tp, _BinaryOper)`
- `template<typename _Iter, typename _Tp, typename _BinaryOper >
_Tp std::__parallel::accumulate (_Iter, _Iter, _Tp, _BinaryOper, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _Tp, typename _BinaryOper >
_Tp std::__parallel::accumulate (_Iter, _Iter, _Tp, _BinaryOper, __gnu_parallel::__Parallelism)`
- `template<typename _Iter, typename _OIter >
_OIter std::__parallel::adjacent_difference (_Iter, _Iter, _OIter)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >
_OIter std::__parallel::adjacent_difference (_Iter, _Iter, _OIter, _BinaryOper)`

- `template<typename _Iter, typename _OIter >`
`_OIter std::__parallel::adjacent_difference (_Iter, _Iter, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >`
`_OIter std::__parallel::adjacent_difference (_Iter, _Iter, _OIter, _BinaryOper, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter >`
`_OIter std::__parallel::adjacent_difference (_Iter, _Iter, _OIter, __gnu_parallel::Parallelism)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >`
`_OIter std::__parallel::adjacent_difference (_Iter, _Iter, _OIter, _BinaryOper, __gnu_parallel::Parallelism)`
- `template<typename _Iter1, typename _Iter2, typename _Tp >`
`_Tp std::__parallel::inner_product (_Iter1, _Iter1, _Iter2, _Tp)`
- `template<typename _Iter1, typename _Iter2, typename _Tp >`
`_Tp std::__parallel::inner_product (_Iter1, _Iter1, _Iter2, _Tp, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Tp >`
`_Tp std::__parallel::inner_product (_Iter1, _Iter1, _Iter2, _Tp, __gnu_parallel::Parallelism)`
- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 >`
`_Tp std::__parallel::inner_product (_Iter1, _Iter1, _Iter2, _Tp, _BinaryFunction1, _BinaryFunction2)`
- `template<typename _Iter1, typename _Iter2, typename _Tp, typename _BinaryFunction1, typename _BinaryFunction2 >`
`_Tp std::__parallel::inner_product (_Iter1, _Iter1, _Iter2, _Tp, _BinaryFunction1, _BinaryFunction2, __gnu_parallel::sequential_tag)`
- `template<typename _Iter1, typename _Iter2, typename _Tp, typename BinaryFunction1, typename BinaryFunction2 >`
`_Tp std::__parallel::inner_product (_Iter1, _Iter1, _Iter2, _Tp, BinaryFunction1, BinaryFunction2, __gnu_parallel::Parallelism)`
- `template<typename _Iter, typename _OIter >`
`_OIter std::__parallel::partial_sum (_Iter, _Iter, _OIter, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >`
`_OIter std::__parallel::partial_sum (_Iter, _Iter, _OIter, _BinaryOper, __gnu_parallel::sequential_tag)`
- `template<typename _Iter, typename _OIter >`
`_OIter std::__parallel::partial_sum (_Iter, _Iter, _OIter __result)`
- `template<typename _Iter, typename _OIter, typename _BinaryOper >`
`_OIter std::__parallel::partial_sum (_Iter, _Iter, _OIter, _BinaryOper)`

6.396.1 Detailed Description

This file is a GNU parallel extension to the Standard C++ Library.

6.397 omp_loop.h File Reference

Namespaces

- [__gnu_parallel](#)

Functions

- `template<typename _RAIter, typename _Op, typename _Fu, typename _Red, typename _Result >`
`_Op __gnu_parallel::__for_each_template_random_access_omp_loop (_RAIter __begin, _RAIter __end, _Op __o, _Fu &__f, _Red __r, _Result __base, _Result &__output, typename std::iterator_traits< _RAIter >::difference_type __bound)`

6.397.1 Detailed Description

Parallelization of embarrassingly parallel execution by means of an OpenMP for loop. This file is a GNU parallel extension to the Standard C++ Library.

6.398 omp_loop_static.h File Reference

Namespaces

- [__gnu_parallel](#)

Functions

- `template<typename _RAIter, typename _Op, typename _Fu, typename _Red, typename _Result >
_Op __gnu_parallel::__for_each_template_random_access_omp_loop_static (_RAIter __begin, _RAIter __end,
_Op __o, _Fu & __f, _Red __r, _Result __base, _Result & __output, typename std::iterator_traits<_RAIter >::difference_type __bound)`

6.398.1 Detailed Description

Parallelization of embarrassingly parallel execution by means of an OpenMP for loop with static scheduling. This file is a GNU parallel extension to the Standard C++ Library.

6.399 opt_random.h File Reference

Namespaces

- [std](#)

6.399.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<random>`.

6.400 optional File Reference

Classes

- `struct std::experimental::fundamentals_v1::_Has_addressof<_Tp >`
- `class std::experimental::fundamentals_v1::_Optional_base<_Tp, _ShouldProvideDestructor >`
- `class std::experimental::fundamentals_v1::_Optional_base<_Tp, false >`
- `class std::experimental::fundamentals_v1::bad_optional_access`
- `struct std::experimental::fundamentals_v1::in_place_t`
- `struct std::experimental::fundamentals_v1::nullopt_t`
- `class std::experimental::fundamentals_v1::optional<_Tp >`
- `class std::experimental::fundamentals_v1::optional<_Tp >`
- `class std::experimental::fundamentals_v1::optional<_Tp >`

Namespaces

- [std](#)

Macros

- `#define __cpp_lib_experimental_optional`
- `#define _GLIBCXX_EXPERIMENTAL_OPTIONAL`

Typedefs

- `template<typename _Tp, typename _Up >`
`using std::experimental::fundamentals_v1::__assigns_from_optional = __or_< is_assignable< _Tp &,`
`const optional< _Up > & >, is_assignable< _Tp &, optional< _Up > & >, is_assignable< _Tp &, const`
`optional< _Up > && >, is_assignable< _Tp &, optional< _Up > && >>`
- `template<typename _Tp, typename _Up >`
`using std::experimental::fundamentals_v1::__converts_from_optional = __or_< is_constructible< _Tp,`
`const optional< _Up > & >, is_constructible< _Tp, optional< _Up > & >, is_constructible< _Tp, const`
`optional< _Up > && >, is_constructible< _Tp, optional< _Up > && >, is_convertible< const optional< _Up`
`> &, _Tp >, is_convertible< optional< _Up > &, _Tp >, is_convertible< const optional< _Up > &&, _Tp >, is_convertible< optional< _Up > &&, _Tp >>`

Functions

- `template<typename _Tp >`
`constexpr enable_if_t<!_Has_addressof< _Tp >::value, _Tp * > std::experimental::fundamentals_v1::__`
`constexpr_addressof (_Tp &__t)`
- `template<typename _Tp >`
`enable_if_t< _Has_addressof< _Tp >::value, _Tp * > std::experimental::fundamentals_v1::__constexpr_`
`addressof (_Tp &__t)`
- `void std::experimental::fundamentals_v1::__throw_bad_optional_access (const char *) __attribute__((__`
`noreturn__))`
- `template<typename _Tp >`
`constexpr optional< decay_t< _Tp > > std::experimental::fundamentals_v1::make_optional (_Tp &&__t)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator!= (const optional< _Tp > &__lhs, const`
`optional< _Tp > &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator!= (const optional< _Tp > &__lhs, nullopt_`
`t) noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator!= (nullopt_t, const optional< _Tp > &__rhs)`
`noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator!= (const optional< _Tp > &__lhs, _Tp const`
`&__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator!= (const _Tp &__lhs, const optional< _Tp >`
`&__rhs)`

- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator< (const optional< _Tp > &__lhs, const optional< _Tp > &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator< (const optional< _Tp > &, nullopt_t) noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator< (nullopt_t, const optional< _Tp > &__rhs) noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator< (const optional< _Tp > &__lhs, const _Tp &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator< (const _Tp &__lhs, const optional< _Tp > &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator<= (const optional< _Tp > &__lhs, const optional< _Tp > &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator<= (const optional< _Tp > &__lhs, nullopt_t) noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator<= (nullopt_t, const optional< _Tp > &) noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator<= (const optional< _Tp > &__lhs, const _Tp &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator<= (const _Tp &__lhs, const optional< _Tp > &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator== (const optional< _Tp > &__lhs, const optional< _Tp > &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator== (const optional< _Tp > &__lhs, nullopt_t) noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator== (nullopt_t, const optional< _Tp > &__rhs) noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator== (const optional< _Tp > &__lhs, const _Tp &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator== (const _Tp &__lhs, const optional< _Tp > &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator> (const optional< _Tp > &__lhs, const optional< _Tp > &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator> (const optional< _Tp > &__lhs, nullopt_t) noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator> (nullopt_t, const optional< _Tp > &) noexcept`

- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator> (const optional< _Tp > &__lhs, const _Tp`
`&__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator> (const _Tp &__lhs, const optional< _Tp >`
`&__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator>= (const optional< _Tp > &__lhs, const`
`optional< _Tp > &__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator>= (const optional< _Tp > &, nullopt_t) noex-`
`cept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator>= (nullopt_t, const optional< _Tp > &__rhs)`
`noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator>= (const optional< _Tp > &__lhs, const _Tp`
`&__rhs)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::operator>= (const _Tp &__lhs, const optional< _Tp >`
`&__rhs)`
- `template<typename _Tp >`
`void std::experimental::fundamentals_v1::swap (optional< _Tp > &__lhs, optional< _Tp > &__rhs)`
`noexcept(noexcept(__lhs.swap(__rhs)))`

Variables

- `constexpr in_place_t std::experimental::fundamentals_v1::in_place`
- `constexpr nullopt_t std::experimental::fundamentals_v1::nullopt`

6.400.1 Detailed Description

This is a TS C++ Library header.

6.400.2 Function Documentation

- 6.400.2.1 `template<typename _Tp > enable_if_t<_Has_addressof<_Tp>::value, _Tp*> std::experimental::fundamentals_v1::↵`
`_constexpr_addressof(_Tp &__t) [inline]`

Fallback overload that defers to `__addressof`.

Definition at line 186 of file optional.

6.401 `order_statistics_imp.hpp` File Reference

6.401.1 Detailed Description

Contains forward declarations for `order_statistics_key`

6.402 order_statistics_imp.hpp File Reference

6.402.1 Detailed Description

Contains forward declarations for order_statistics_key

6.403 ordered_base.h File Reference

Namespaces

- [std](#)
- [std::__profile](#)

6.403.1 Detailed Description

This file is a GNU profile extension to the Standard C++ Library.

6.404 os_defines.h File Reference

Macros

- `#define __NO_CTYPE`
- `#define _GLIBCXX_NO_OBSOLETE_ISINF_ISNAN_DYNAMIC`

6.404.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iosfwd>`.

6.405 ostream File Reference

Classes

- class [std::basic_ostream<_CharT, _Traits>](#)
- class [std::basic_ostream<_CharT, _Traits>::sentry](#)

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_OSTREAM`

Functions

- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::endl (basic_ostream< _CharT, _Traits > &__os)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::ends (basic_ostream< _CharT, _Traits > &__os)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::flush (basic_ostream< _CharT, _Traits > &__os)`
- `template<typename _CharT, typename _Traits, typename _Tp >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, const _Tp &__x)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__out, _CharT __c)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__out, char __c)`
- `template<class _Traits >`
`basic_ostream< char, _Traits > & std::operator<< (basic_ostream< char, _Traits > &__out, char __c)`
- `template<class _Traits >`
`basic_ostream< char, _Traits > & std::operator<< (basic_ostream< char, _Traits > &__out, signed char __c)`
- `template<class _Traits >`
`basic_ostream< char, _Traits > & std::operator<< (basic_ostream< char, _Traits > &__out, unsigned char __c)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__out, const _CharT *__s)`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__out, const char *__s)`
- `template<class _Traits >`
`basic_ostream< char, _Traits > & std::operator<< (basic_ostream< char, _Traits > &__out, const char *__s)`
- `template<class _Traits >`
`basic_ostream< char, _Traits > & std::operator<< (basic_ostream< char, _Traits > &__out, const signed char *__s)`
- `template<class _Traits >`
`basic_ostream< char, _Traits > & std::operator<< (basic_ostream< char, _Traits > &__out, const unsigned char *__s)`

6.405.1 Detailed Description

This is a Standard C++ Library header.

6.406 ostream.tcc File Reference

Namespaces

- [std](#)

Macros

- `#define _OSTREAM_TCC`

Functions

- `template<typename _CharT, typename _Traits >
basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__out, const char
*__s)`

6.406.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ostream>`.

6.407 ostream_insert.h File Reference

Namespaces

- [std](#)

Functions

- `template<typename _CharT, typename _Traits >
void std::__ostream_fill (basic_ostream< _CharT, _Traits > &__out, streamsize __n)`
- `template<typename _CharT, typename _Traits >
basic_ostream< _CharT, _Traits > & std::__ostream_insert (basic_ostream< _CharT, _Traits > &__out, const
_CharT *__s, streamsize __n)`
- `template<typename _CharT, typename _Traits >
void std::__ostream_write (basic_ostream< _CharT, _Traits > &__out, const _CharT *__s, streamsize __n)`

6.407.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ostream>`.

6.408 `ov_tree_map.hpp` File Reference

Classes

- class [__gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >](#)
- class [__gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >::cond_dtor< Size_Type >](#)

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CONST_NODE_ITERATOR_NAME`
- `#define PB_DS_OV_TREE_NAME`
- `#define PB_DS_OV_TREE_TRAITS_BASE`

6.408.1 Detailed Description

Contains an implementation class for `ov_tree`.

6.409 `pairing_heap.hpp` File Reference

Classes

- class [__gnu_pbds::detail::pairing_heap< Value_Type, Cmp_Fn, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_ASSERT_NODE_CONSISTENT(_Node, _Bool)`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_P_HEAP_BASE`

6.409.1 Detailed Description

Contains an implementation class for a pairing heap.

6.410 par_loop.h File Reference

Namespaces

- [__gnu_parallel](#)

Functions

- `template<typename _RAIter, typename _Op, typename _Fu, typename _Red, typename _Result >
_Op __gnu_parallel::__for_each_template_random_access_ed (_RAIter __begin, _RAIter __end, _Op __o, _Fu
& __f, _Red __r, _Result __base, _Result & __output, typename std::iterator_traits< _RAIter >::difference_type
__bound)`

6.410.1 Detailed Description

Parallelization of embarrassingly parallel execution by means of equal splitting. This file is a GNU parallel extension to the Standard C++ Library.

6.411 parallel.h File Reference

6.411.1 Detailed Description

End-user include file. Provides advanced settings and tuning options. This file is a GNU parallel extension to the Standard C++ Library.

6.412 parse_numbers.h File Reference

Namespaces

- [std](#)

Typedefs

- `template<unsigned long long _Val>
using std::__parse_int::__ull_constant = integral_constant< unsigned long long, _Val >`
- `template<char... _Digs>
using std::__select_int::__Select_int = typename _Select_int_base< __parse_int::__Parse_int< _Digs... >↔
::value, unsigned char, unsigned short, unsigned int, unsigned long, unsigned long long >::type`

6.412.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<chrono>`.

6.413 `partial_sum.h` File Reference

Namespaces

- [__gnu_parallel](#)

Functions

- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator __gnu_parallel::__parallel_partial_sum (_Iter __begin, _Iter __end, _OutputIterator __result, \leftrightarrow`
`_BinaryOperation __bin_op)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator __gnu_parallel::__parallel_partial_sum_basecase (_Iter __begin, _Iter __end, _OutputIterator`
`__result, _BinaryOperation __bin_op, typename std::iterator_traits< _Iter >::value_type __value)`
- `template<typename _Iter, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator __gnu_parallel::__parallel_partial_sum_linear (_Iter __begin, _Iter __end, _OutputIterator __ \leftrightarrow`
`result, _BinaryOperation __bin_op, typename std::iterator_traits< _Iter >::difference_type __n)`

6.413.1 Detailed Description

Parallel implementation of `std::partial_sum()`, i.e. prefix sums. This file is a GNU parallel extension to the Standard C++ Library.

6.414 `partition.h` File Reference

Namespaces

- [__gnu_parallel](#)

Macros

- `#define _GLIBCXX_VOLATILE`

Functions

- `template<typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_nth_element (_RAIter __begin, _RAIter __nth, _RAIter __end, _Compare __ \leftrightarrow`
`comp)`
- `template<typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_partial_sort (_RAIter __begin, _RAIter __middle, _RAIter __end, _Compare __ \leftrightarrow`
`comp)`
- `template<typename _RAIter, typename _Predicate >`
`std::iterator_traits< _RAIter >::difference_type __gnu_parallel::__parallel_partition (_RAIter __begin, _RAIter \leftrightarrow`
`__end, _Predicate __pred, _ThreadIndex __num_threads)`

6.414.1 Detailed Description

Parallel implementation of `std::partition()`, `std::nth_element()`, and `std::partial_sort()`. This file is a GNU parallel extension to the Standard C++ Library.

6.414.2 Macro Definition Documentation

6.414.2.1 `#define _GLIBCXX_VOLATILE`

Decide whether to declare certain variables volatile.

Definition at line 43 of file `partition.h`.

Referenced by `__gnu_parallel::__parallel_partition()`.

6.415 pat_trie_.hpp File Reference

Classes

- class [__gnu_pbds::detail::pat_trie_map< Key, Mapped, Node_And_It_Traits, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_ASSERT_NODE_VALID(X)`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_PAT_TRIE_NAME`
- `#define PB_DS_PAT_TRIE_TRAITS_BASE`
- `#define PB_DS_RECURSIVE_COUNT_LEAFS(X)`

6.415.1 Detailed Description

Contains an implementation class for a patricia tree.

6.416 pat_trie_base.hpp File Reference

Classes

- struct [__gnu_pbds::detail::pat_trie_base](#)
- class [__gnu_pbds::detail::pat_trie_base::_Clter< Node, Leaf, Head, Inode, Is_Forward_Iterator >](#)
- struct [__gnu_pbds::detail::pat_trie_base::_Head< _ATraits, Metadata >](#)
- struct [__gnu_pbds::detail::pat_trie_base::_Inode< _ATraits, Metadata >](#)
- struct [__gnu_pbds::detail::pat_trie_base::_Inode< _ATraits, Metadata >::const_iterator](#)
- struct [__gnu_pbds::detail::pat_trie_base::_Inode< _ATraits, Metadata >::iterator](#)
- class [__gnu_pbds::detail::pat_trie_base::_Iter< Node, Leaf, Head, Inode, Is_Forward_Iterator >](#)
- struct [__gnu_pbds::detail::pat_trie_base::_Leaf< _ATraits, Metadata >](#)
- struct [__gnu_pbds::detail::pat_trie_base::_Metadata< Metadata, _Alloc >](#)
- struct [__gnu_pbds::detail::pat_trie_base::_Metadata< null_type, _Alloc >](#)
- struct [__gnu_pbds::detail::pat_trie_base::_Node_base< _ATraits, Metadata >](#)
- class [__gnu_pbds::detail::pat_trie_base::_Node_citer< Node, Leaf, Head, Inode, _Clterator, Iterator, _Alloc >](#)
- class [__gnu_pbds::detail::pat_trie_base::_Node_iter< Node, Leaf, Head, Inode, _Clterator, Iterator, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

Macros

- **#define PB_DS_CLASS_C_DEC**
- **#define PB_DS_CLASS_T_DEC**
- **#define PB_DS_CONST_IT_C_DEC**
- **#define PB_DS_CONST_ODIR_IT_C_DEC**
- **#define PB_DS_IT_C_DEC**
- **#define PB_DS_ODIR_IT_C_DEC**
- **#define PB_DS_PAT_TRIE_NODE_CONST_ITERATOR_C_DEC**
- **#define PB_DS_PAT_TRIE_NODE_ITERATOR_C_DEC**

6.416.1 Detailed Description

Contains the base class for a patricia tree.

6.417 pod_char_traits.h File Reference

Classes

- struct [__gnu_cxx::character< _Value, _Int, _St >](#)
- struct [std::char_traits< __gnu_cxx::character< _Value, _Int, _St > >](#)

Namespaces

- [__gnu_cxx](#)
- [std](#)

Functions

- `template<typename _Value, typename _Int, typename _St >
bool __gnu_cxx::operator< (const character< _Value, _Int, _St > &lhs, const character< _Value, _Int, _St > &rhs)`
- `template<typename _Value, typename _Int, typename _St >
bool __gnu_cxx::operator== (const character< _Value, _Int, _St > &lhs, const character< _Value, _Int, _St > &rhs)`

6.417.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.418 point_const_iterator.hpp File Reference

Classes

- class [__gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

6.418.1 Detailed Description

Contains an iterator class returned by the table's const find and insert methods.

6.419 point_const_iterator.hpp File Reference

Classes

- class [__gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_< Node, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`

6.419.1 Detailed Description

Contains an iterator class returned by the table's const find and insert methods.

6.420 `point_const_iterator.hpp` File Reference

Classes

- class [point_const_iterator_](#)

6.420.1 Detailed Description

Contains an iterator class returned by the tables' const find and insert methods.

6.421 `point_iterator.hpp` File Reference

Classes

- class [point_iterator_](#)

6.421.1 Detailed Description

Contains an iterator class returned by the tables' find and insert methods.

6.422 `point_iterators.hpp` File Reference

Classes

- class [__gnu_pbds::detail::bin_search_tree_const_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >](#)
- class [__gnu_pbds::detail::bin_search_tree_it_< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_TREE_CONST_IT_C_DEC`
- `#define PB_DS_TREE_CONST_ODIR_IT_C_DEC`
- `#define PB_DS_TREE_IT_C_DEC`
- `#define PB_DS_TREE_ODIR_IT_C_DEC`

6.422.1 Detailed Description

Contains an implementation class for `bin_search_tree_`.

6.423 pointer.h File Reference

Classes

- struct [__gnu_cxx::_Invalid_type](#)
- class [__gnu_cxx::_Pointer_adapter< _Storage_policy >](#)
- class [__gnu_cxx::_Relative_pointer_impl< _Tp >](#)
- class [__gnu_cxx::_Relative_pointer_impl< const _Tp >](#)
- class [__gnu_cxx::_Std_pointer_impl< _Tp >](#)
- struct [__gnu_cxx::_Unqualified_type< _Tp >](#)

Namespaces

- [__gnu_cxx](#)
- [std](#)

Macros

- `#define CXX_POINTER_ARITH_OPERATOR_SET(INT_TYPE)`
- `#define GCC_CXX_POINTER_COMPARISON_OPERATION_SET(OPERATOR)`

Functions

- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator!= (const _Pointer_adapter< _Tp1 > &__lhs, _Tp2 __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator!= (_Tp1 __lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator!= (const _Pointer_adapter< _Tp1 > &__lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp >`
`bool __gnu_cxx::operator!= (const _Pointer_adapter< _Tp > &__lhs, int __rhs)`
- `template<typename _Tp >`
`bool __gnu_cxx::operator!= (int __lhs, const _Pointer_adapter< _Tp > &__rhs)`

- `template<typename _Tp >`
`bool __gnu_cxx::operator!= (const _Pointer_adapter< _Tp > &__lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator< (_Tp1 __lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator< (const _Pointer_adapter< _Tp1 > &__lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator< (const _Pointer_adapter< _Tp1 > &__lhs, _Tp2 __rhs)`
- `template<typename _CharT, typename _Traits, typename _StoreT >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const _Pointer_adapter< _StoreT > &__p)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator<= (const _Pointer_adapter< _Tp1 > &__lhs, _Tp2 __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator<= (_Tp1 __lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator<= (const _Pointer_adapter< _Tp1 > &__lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp >`
`bool __gnu_cxx::operator<= (const _Pointer_adapter< _Tp > &__lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator== (_Tp1 __lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator== (const _Pointer_adapter< _Tp1 > &__lhs, _Tp2 __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator== (const _Pointer_adapter< _Tp1 > &__lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp >`
`bool __gnu_cxx::operator== (const _Pointer_adapter< _Tp > &__lhs, int __rhs)`
- `template<typename _Tp >`
`bool __gnu_cxx::operator== (int __lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<typename _Tp >`
`bool __gnu_cxx::operator== (const _Pointer_adapter< _Tp > &__lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator> (_Tp1 __lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator> (const _Pointer_adapter< _Tp1 > &__lhs, _Tp2 __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator> (const _Pointer_adapter< _Tp1 > &__lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp >`
`bool __gnu_cxx::operator> (const _Pointer_adapter< _Tp > &__lhs, const _Pointer_adapter< _Tp > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator>= (const _Pointer_adapter< _Tp1 > &__lhs, _Tp2 __rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator>= (const _Pointer_adapter< _Tp1 > &__lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp1, typename _Tp2 >`
`bool __gnu_cxx::operator>= (_Tp1 __lhs, const _Pointer_adapter< _Tp2 > &__rhs)`
- `template<typename _Tp >`
`bool __gnu_cxx::operator>= (const _Pointer_adapter< _Tp > &__lhs, const _Pointer_adapter< _Tp > &__rhs)`

6.423.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

Author

Bob Walters

Provides reusable `_Pointer_adapter` for assisting in the development of custom pointer types that can be used with the standard containers via the `allocator::pointer` and `allocator::const_pointer` typedefs.

6.424 policy_access_fn_imps.hpp File Reference

6.424.1 Detailed Description

Contains an implementation class for a `binary_heap`.

6.425 policy_access_fn_imps.hpp File Reference

6.425.1 Detailed Description

Contains an implementation class for `bin_search_tree_`.

6.426 policy_access_fn_imps.hpp File Reference

6.426.1 Detailed Description

Contains implementations of `cc_ht_map_`'s policy access functions.

6.427 policy_access_fn_imps.hpp File Reference

6.427.1 Detailed Description

Contains implementations of `gp_ht_map_`'s policy access functions.

6.428 policy_access_fn_imps.hpp File Reference

6.428.1 Detailed Description

Contains an implementation class for `left_child_next_sibling_heap_`.

6.429 `policy_access_fn_imps.hpp` File Reference

6.429.1 Detailed Description

Contains an implementation class for `ov_tree`.

6.430 `policy_access_fn_imps.hpp` File Reference

6.430.1 Detailed Description

Contains an implementation class for `pat_trie`.

6.431 `pool_allocator.h` File Reference

Classes

- class [__gnu_cxx::__pool_alloc< _Tp >](#)
- class [__gnu_cxx::__pool_alloc_base](#)

Namespaces

- [__gnu_cxx](#)

Functions

- `template<typename _Tp >`
`bool __gnu_cxx::operator!= (const __pool_alloc< _Tp > &, const __pool_alloc< _Tp > &)`
- `template<typename _Tp >`
`bool __gnu_cxx::operator== (const __pool_alloc< _Tp > &, const __pool_alloc< _Tp > &)`

6.431.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.432 `postypes.h` File Reference

Classes

- class [std::fpos< _StateT >](#)

Namespaces

- [std](#)

Typedefs

- typedef long long [std::streamoff](#)
- typedef fpos< mbstate_t > [std::streampos](#)
- typedef ptrdiff_t [std::streamsize](#)
- typedef fpos< mbstate_t > [std::u16streampos](#)
- typedef fpos< mbstate_t > [std::u32streampos](#)
- typedef fpos< mbstate_t > [std::wstreampos](#)

Functions

- template<typename _StateT >
bool **std::operator!=** (const fpos< _StateT > &__lhs, const fpos< _StateT > &__rhs)
- template<typename _StateT >
bool [std::operator==](#) (const fpos< _StateT > &__lhs, const fpos< _StateT > &__rhs)

6.432.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iosfwd>`.

6.433 predefined_ops.h File Reference

Namespaces

- [__gnu_cxx](#)

Functions

- template<typename _Compare >
`_GLIBCXX14_CONSTEXPR` `_Iter_comp_iter< _Compare > __gnu_cxx::__ops::__iter_comp_iter (_Compare __comp)`
- template<typename _Iterator >
`_Iter_equals_iter< _Iterator > __gnu_cxx::__ops::__iter_comp_iter (_Iter_equal_to_iter, _Iterator __it)`
- template<typename _Compare, typename _Iterator >
`_Iter_comp_to_iter< _Compare, _Iterator > __gnu_cxx::__ops::__iter_comp_iter (_Iter_comp_iter< _Compare > __comp, _Iterator __it)`
- `_Iter_less_val __gnu_cxx::__ops::__iter_comp_val (_Iter_less_iter)`
- `_Iter_equal_to_val __gnu_cxx::__ops::__iter_comp_val (_Iter_equal_to_iter)`
- template<typename _Compare >
`_Iter_comp_val< _Compare > __gnu_cxx::__ops::__iter_comp_val (_Compare __comp)`

- `template<typename _Compare >`
`_lter_comp_val< _Compare > __gnu_cxx::__ops::__lter_comp_val (_lter_comp_iter< _Compare > __comp)`
- `template<typename _Compare, typename _Value >`
`_lter_comp_to_val< _Compare, _Value > __gnu_cxx::__ops::__lter_comp_val (_Compare __comp, _Value &__val)`
- `_lter_equal_to_iter __gnu_cxx::__ops::__lter_equal_to_iter ()`
- `_lter_equal_to_val __gnu_cxx::__ops::__lter_equal_to_val ()`
- `template<typename _Value >`
`_lter_equals_val< _Value > __gnu_cxx::__ops::__lter_equals_val (_Value &__val)`
- `_GLIBCXX14_CONSTEXPR _lter_less_iter __gnu_cxx::__ops::__lter_less_iter ()`
- `_lter_less_val __gnu_cxx::__ops::__lter_less_val ()`
- `template<typename _Predicate >`
`_lter_negate< _Predicate > __gnu_cxx::__ops::__negate (_lter_pred< _Predicate > __pred)`
- `template<typename _Predicate >`
`_lter_pred< _Predicate > __gnu_cxx::__ops::__pred_iter (_Predicate __pred)`
- `_Val_less_iter __gnu_cxx::__ops::__val_comp_iter (_lter_less_iter)`
- `template<typename _Compare >`
`_Val_comp_iter< _Compare > __gnu_cxx::__ops::__val_comp_iter (_Compare __comp)`
- `template<typename _Compare >`
`_Val_comp_iter< _Compare > __gnu_cxx::__ops::__val_comp_iter (_lter_comp_iter< _Compare > __comp)`
- `_Val_less_iter __gnu_cxx::__ops::__val_less_iter ()`

6.433.1 Detailed Description

This is an internal header file, included by other library headers. You should not attempt to use it directly.

6.434 prefix_search_node_update_imp.hpp File Reference

6.434.1 Detailed Description

Contains an implementation of `prefix_search_node_update`.

6.435 priority_queue.hpp File Reference

Classes

- class [`__gnu_pbds::priority_queue< _Tv, Cmp_Fn, Tag, _Alloc >`](#)

Namespaces

- [`__gnu_pbds`](#)

6.435.1 Detailed Description

Contains `priority_queues`.

6.436 `priority_queue_base_dispatch.hpp` File Reference

Classes

- struct [__gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, binary_heap_tag, null_type>](#)
- struct [__gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, binomial_heap_tag, null_type>](#)
- struct [__gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, pairing_heap_tag, null_type>](#)
- struct [__gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, rc_binomial_heap_tag, null_type>](#)
- struct [__gnu_pbds::detail::container_base_dispatch<_VTp, Cmp_Fn, _Alloc, thin_heap_tag, null_type>](#)

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_ASSERT_VALID(X)`
- `#define PB_DS_DEBUG_VERIFY(_Cond)`

6.436.1 Detailed Description

Contains an pqiative container dispatching base.

6.437 `probe_fn_base.hpp` File Reference

Classes

- class [__gnu_pbds::detail::probe_fn_base<_Alloc>](#)

Namespaces

- [__gnu_pbds](#)

6.437.1 Detailed Description

Contains a probe policy base.

6.438 `profiler.h` File Reference

Classes

- struct [__gnu_profile::__reentrance_guard](#)

Namespaces

- [__gnu_profile](#)

Macros

- #define **__profcxx_hash_func_construct**(__x...)
- #define **__profcxx_hash_func_destruct**(__x...)
- #define **__profcxx_hashtable_size_construct**(__x...)
- #define **__profcxx_hashtable_size_destruct**(__x...)
- #define **__profcxx_hashtable_size_resize**(__x...)
- #define **__profcxx_is_invalid**()
- #define **__profcxx_is_off**()
- #define **__profcxx_is_on**()
- #define **__profcxx_list2slist_construct**(__x...)
- #define **__profcxx_list2slist_destruct**(__x...)
- #define **__profcxx_list2slist_operation**(__x...)
- #define **__profcxx_list2slist_rewind**(__x...)
- #define **__profcxx_list2vector_construct**(__x...)
- #define **__profcxx_list2vector_destruct**(__x...)
- #define **__profcxx_list2vector_insert**(__x...)
- #define **__profcxx_list2vector_invalid_operator**(__x...)
- #define **__profcxx_list2vector_iterate**(__x...)
- #define **__profcxx_map2umap_construct**(__x...)
- #define **__profcxx_map2umap_destruct**(__x...)
- #define **__profcxx_map2umap_erase**(__x...)
- #define **__profcxx_map2umap_find**(__x...)
- #define **__profcxx_map2umap_insert**(__x...)
- #define **__profcxx_map2umap_invalidate**(__x...)
- #define **__profcxx_map2umap_iterate**(__x...)
- #define **__profcxx_report**()
- #define **__profcxx_turn_off**()
- #define **__profcxx_turn_on**()
- #define **__profcxx_vector2list_construct**(__x...)
- #define **__profcxx_vector2list_destruct**(__x...)
- #define **__profcxx_vector2list_insert**(__x...)
- #define **__profcxx_vector2list_invalid_operator**(__x...)
- #define **__profcxx_vector2list_iterate**(__x...)
- #define **__profcxx_vector2list_resize**(__x...)
- #define **__profcxx_vector_size_construct**(__x...)
- #define **__profcxx_vector_size_destruct**(__x...)
- #define **__profcxx_vector_size_resize**(__x...)
- #define **GLIBCXX_PROFILE_DATA**(__name)
- #define **GLIBCXX_PROFILE_DEFINE_DATA**(__type, __name, __initial_value...)
- #define **GLIBCXX_PROFILE_DEFINE_UNINIT_DATA**(__type, __name)
- #define **GLIBCXX_PROFILE_MAX_STACK_DEPTH**
- #define **GLIBCXX_PROFILE_MAX_STACK_DEPTH_ENV_VAR**
- #define **GLIBCXX_PROFILE_MAX_WARN_COUNT**
- #define **GLIBCXX_PROFILE_MAX_WARN_COUNT_ENV_VAR**
- #define **GLIBCXX_PROFILE_MEM_PER_DIAGNOSTIC**
- #define **GLIBCXX_PROFILE_MEM_PER_DIAGNOSTIC_ENV_VAR**
- #define **GLIBCXX_PROFILE_TRACE_ENV_VAR**
- #define **GLIBCXX_PROFILE_TRACE_PATH_ROOT**

Functions

- `bool __gnu_profile::__is_invalid ()`
- `bool __gnu_profile::__is_off ()`
- `bool __gnu_profile::__is_on ()`
- `void __gnu_profile::__report ()`
- `__hashfunc_info * __gnu_profile::__trace_hash_func_construct ()`
- `void __gnu_profile::__trace_hash_func_destruct (__hashfunc_info *, std::size_t, std::size_t, std::size_t)`
- `__container_size_info * __gnu_profile::__trace_hashtable_size_construct (std::size_t)`
- `void __gnu_profile::__trace_hashtable_size_destruct (__container_size_info *, std::size_t, std::size_t)`
- `void __gnu_profile::__trace_hashtable_size_resize (__container_size_info *, std::size_t, std::size_t)`
- `__list2slist_info * __gnu_profile::__trace_list_to_slist_construct ()`
- `void __gnu_profile::__trace_list_to_slist_destruct (__list2slist_info *)`
- `void __gnu_profile::__trace_list_to_slist_operation (__list2slist_info *)`
- `void __gnu_profile::__trace_list_to_slist_rewind (__list2slist_info *)`
- `__list2vector_info * __gnu_profile::__trace_list_to_vector_construct ()`
- `void __gnu_profile::__trace_list_to_vector_destruct (__list2vector_info *)`
- `void __gnu_profile::__trace_list_to_vector_insert (__list2vector_info *, std::size_t, std::size_t)`
- `void __gnu_profile::__trace_list_to_vector_invalid_operator (__list2vector_info *)`
- `void __gnu_profile::__trace_list_to_vector_iterate (__list2vector_info *, int)`
- `void __gnu_profile::__trace_list_to_vector_resize (__list2vector_info *, std::size_t, std::size_t)`
- `__map2umap_info * __gnu_profile::__trace_map_to_unordered_map_construct ()`
- `void __gnu_profile::__trace_map_to_unordered_map_destruct (__map2umap_info *)`
- `void __gnu_profile::__trace_map_to_unordered_map_erase (__map2umap_info *, std::size_t, std::size_t)`
- `void __gnu_profile::__trace_map_to_unordered_map_find (__map2umap_info *, std::size_t)`
- `void __gnu_profile::__trace_map_to_unordered_map_insert (__map2umap_info *, std::size_t, std::size_t)`
- `void __gnu_profile::__trace_map_to_unordered_map_invalidate (__map2umap_info *)`
- `void __gnu_profile::__trace_map_to_unordered_map_iterate (__map2umap_info *, std::size_t)`
- `__container_size_info * __gnu_profile::__trace_vector_size_construct (std::size_t)`
- `void __gnu_profile::__trace_vector_size_destruct (__container_size_info *, std::size_t, std::size_t)`
- `void __gnu_profile::__trace_vector_size_resize (__container_size_info *, std::size_t, std::size_t)`
- `__vector2list_info * __gnu_profile::__trace_vector_to_list_construct ()`
- `void __gnu_profile::__trace_vector_to_list_destruct (__vector2list_info *)`
- `void __gnu_profile::__trace_vector_to_list_insert (__vector2list_info *, std::size_t, std::size_t)`
- `void __gnu_profile::__trace_vector_to_list_invalid_operator (__vector2list_info *)`
- `void __gnu_profile::__trace_vector_to_list_iterate (__vector2list_info *, int)`
- `void __gnu_profile::__trace_vector_to_list_resize (__vector2list_info *, std::size_t, std::size_t)`
- `bool __gnu_profile::__turn_off ()`
- `bool __gnu_profile::__turn_on ()`

6.438.1 Detailed Description

Interface of the profiling runtime library.

6.439 profiler_algos.h File Reference

Namespaces

- [__gnu_profile](#)

Functions

- `template<typename _InputIterator, typename _Function >`
`_Function __gnu_profile::__for_each (_InputIterator __first, _InputIterator __last, _Function __f)`
- `template<typename _Container >`
`void __gnu_profile::__insert_top_n (_Container &__output, const typename _Container::value_type &__value,`
`typename _Container::size_type __n)`
- `template<typename _ForwardIterator, typename _Tp >`
`_ForwardIterator __gnu_profile::__remove (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__↵`
`value)`
- `template<typename _Container >`
`void __gnu_profile::__top_n (const _Container &__input, _Container &__output, typename _Container::size_↵`
`type __n)`

6.439.1 Detailed Description

Algorithms used by the profile extension.

This file is needed to avoid including `<algorithm>` or `<bits/stl_algo.h>`. Including those files would result in recursive includes. These implementations are oversimplified. In general, efficiency may be sacrificed to minimize maintenance overhead.

6.440 profiler_container_size.h File Reference

Classes

- class [`__gnu_profile::__container_size_info`](#)
- class [`__gnu_profile::__container_size_stack_info`](#)
- class [`__gnu_profile::__trace_container_size`](#)

Namespaces

- [`__gnu_profile`](#)

6.440.1 Detailed Description

Diagnostics for container sizes.

6.441 profiler_hash_func.h File Reference

Classes

- class [`__gnu_profile::__hashfunc_info`](#)
- class [`__gnu_profile::__hashfunc_stack_info`](#)
- class [`__gnu_profile::__trace_hash_func`](#)

Namespaces

- [__gnu_profile](#)

Functions

- `__hashfunc_info * __gnu_profile::__trace_hash_func_construct ()`
- `void __gnu_profile::__trace_hash_func_destruct (__hashfunc_info *, std::size_t, std::size_t, std::size_t)`
- `void __gnu_profile::__trace_hash_func_free ()`
- `void __gnu_profile::__trace_hash_func_init ()`
- `void __gnu_profile::__trace_hash_func_report (FILE * __f, __warning_vector_t & __warnings)`

6.441.1 Detailed Description

Data structures to represent profiling traces.

6.442 profiler_hashtable_size.h File Reference

Classes

- class [__gnu_profile::__trace_hashtable_size](#)

Namespaces

- [__gnu_profile](#)

Functions

- `__container_size_info * __gnu_profile::__trace_hashtable_size_construct (std::size_t)`
- `void __gnu_profile::__trace_hashtable_size_destruct (__container_size_info *, std::size_t, std::size_t)`
- `void __gnu_profile::__trace_hashtable_size_free ()`
- `void __gnu_profile::__trace_hashtable_size_init ()`
- `void __gnu_profile::__trace_hashtable_size_report (FILE * __f, __warning_vector_t & __warnings)`
- `void __gnu_profile::__trace_hashtable_size_resize (__container_size_info *, std::size_t, std::size_t)`

6.442.1 Detailed Description

Collection of hashtable size traces.

6.443 profiler_list_to_slist.h File Reference

Namespaces

- [__gnu_profile](#)

Functions

- `__list2slist_info * __gnu_profile::__trace_list_to_slist_construct ()`
- `void __gnu_profile::__trace_list_to_slist_destruct (__list2slist_info *)`
- `void __gnu_profile::__trace_list_to_slist_free ()`
- `void __gnu_profile::__trace_list_to_slist_init ()`
- `void __gnu_profile::__trace_list_to_slist_operation (__list2slist_info *)`
- `void __gnu_profile::__trace_list_to_slist_report (FILE *__f, __warning_vector_t &__warnings)`
- `void __gnu_profile::__trace_list_to_slist_rewind (__list2slist_info *)`

6.443.1 Detailed Description

Diagnostics for list to slist.

6.444 profiler_list_to_vector.h File Reference

Classes

- class [__gnu_profile::__list2vector_info](#)

Namespaces

- [__gnu_profile](#)

Functions

- `__list2vector_info * __gnu_profile::__trace_list_to_vector_construct ()`
- `void __gnu_profile::__trace_list_to_vector_destruct (__list2vector_info *)`
- `void __gnu_profile::__trace_list_to_vector_free ()`
- `void __gnu_profile::__trace_list_to_vector_init ()`
- `void __gnu_profile::__trace_list_to_vector_insert (__list2vector_info *, std::size_t, std::size_t)`
- `void __gnu_profile::__trace_list_to_vector_invalid_operator (__list2vector_info *)`
- `void __gnu_profile::__trace_list_to_vector_iterate (__list2vector_info *, int)`
- `void __gnu_profile::__trace_list_to_vector_report (FILE *__f, __warning_vector_t &__warnings)`
- `void __gnu_profile::__trace_list_to_vector_resize (__list2vector_info *, std::size_t, std::size_t)`

6.444.1 Detailed Description

diagnostics for list to vector.

6.445 profiler_map_to_unordered_map.h File Reference

Classes

- class [__gnu_profile::__map2umap_info](#)
- class [__gnu_profile::__map2umap_stack_info](#)
- class [__gnu_profile::__trace_map2umap](#)

Namespaces

- [__gnu_profile](#)

Functions

- int [__gnu_profile::__log2](#) (std::size_t __size)
- float [__gnu_profile::__map_erase_cost](#) (std::size_t __size)
- float [__gnu_profile::__map_find_cost](#) (std::size_t __size)
- float [__gnu_profile::__map_insert_cost](#) (std::size_t __size)
- [__map2umap_info](#) * [__gnu_profile::__trace_map_to_unordered_map_construct](#) ()
- void [__gnu_profile::__trace_map_to_unordered_map_destruct](#) ([__map2umap_info](#) *)
- void [__gnu_profile::__trace_map_to_unordered_map_erase](#) ([__map2umap_info](#) *, std::size_t, std::size_t)
- void [__gnu_profile::__trace_map_to_unordered_map_find](#) ([__map2umap_info](#) *, std::size_t)
- void [__gnu_profile::__trace_map_to_unordered_map_free](#) ()
- void [__gnu_profile::__trace_map_to_unordered_map_init](#) ()
- void [__gnu_profile::__trace_map_to_unordered_map_insert](#) ([__map2umap_info](#) *, std::size_t, std::size_t)
- void [__gnu_profile::__trace_map_to_unordered_map_invalidate](#) ([__map2umap_info](#) *)
- void [__gnu_profile::__trace_map_to_unordered_map_iterate](#) ([__map2umap_info](#) * __info, int)
- void [__gnu_profile::__trace_map_to_unordered_map_report](#) (FILE * __f, [__warning_vector_t](#) & __warnings)

6.445.1 Detailed Description

Diagnostics for map to unordered_map.

6.446 profiler_node.h File Reference

Classes

- class [__gnu_profile::__object_info_base](#)
- class [__gnu_profile::__stack_hash](#)

Namespaces

- [__gnu_profile](#)

Typedefs

- typedef void * **__gnu_profile::__instruction_address_t**
- typedef std::vector< __instruction_address_t > **__gnu_profile::__stack_npt**
- typedef __stack_npt * **__gnu_profile::__stack_t**

Functions

- **__stack_t __gnu_profile::__get_stack ()**
- **std::size_t __gnu_profile::__size (__stack_t __stack)**
- **std::size_t __gnu_profile::__stack_max_depth ()**
- **void __gnu_profile::__write (FILE * __f, __stack_t __stack)**

6.446.1 Detailed Description

Data structures to represent a single profiling event.

6.447 profiler_state.h File Reference

Namespaces

- [**__gnu_profile**](#)

Enumerations

- enum **__state_type** { **__ON**, **__OFF**, **__INVALID** }

Functions

- **bool __gnu_profile::__is_invalid ()**
- **bool __gnu_profile::__is_off ()**
- **bool __gnu_profile::__is_on ()**
- **bool __gnu_profile::__turn (__state_type __s)**
- **bool __gnu_profile::__turn_off ()**
- **bool __gnu_profile::__turn_on ()**
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA (__state_type, __state, __INVALID)**

6.447.1 Detailed Description

Global profiler state.

6.448 profiler_trace.h File Reference

Classes

- class [__gnu_profile::__trace_base< __object_info, __stack_info >](#)
- struct [__gnu_profile::__warning_data](#)

Namespaces

- [__gnu_profile](#)

Macros

- #define [_GLIBCXX_IMPL_UNORDERED_MAP](#)

Typedefs

- typedef std::vector< __cost_factor * > [__gnu_profile::__cost_factor_vector](#)
- typedef std::unordered_map< [std::string](#), [std::string](#) > [__gnu_profile::__env_t](#)
- typedef std::vector< __warning_data > [__gnu_profile::__warning_vector_t](#)

Functions

- std::size_t [__gnu_profile::__env_to_size_t](#) (const char * __env_var, std::size_t __default_value)
- int [__gnu_profile::__log_magnitude](#) (float __f)
- std::size_t [__gnu_profile::__max_mem](#) ()
- FILE * [__gnu_profile::__open_output_file](#) (const char * __extension)
- bool [__gnu_profile::__profcxx_init](#) ()
- void [__gnu_profile::__profcxx_init_unconditional](#) ()
- void [__gnu_profile::__read_cost_factors](#) ()
- void [__gnu_profile::__report](#) ()
- void [__gnu_profile::__report_and_free](#) ()
- void [__gnu_profile::__set_cost_factors](#) ()
- void [__gnu_profile::__set_max_mem](#) ()
- void [__gnu_profile::__set_max_stack_trace_depth](#) ()
- void [__gnu_profile::__set_max_warn_count](#) ()
- void [__gnu_profile::__set_trace_path](#) ()
- std::size_t [__gnu_profile::__stack_max_depth](#) ()
- void [__gnu_profile::__trace_hash_func_free](#) ()
- void [__gnu_profile::__trace_hash_func_init](#) ()
- void [__gnu_profile::__trace_hash_func_report](#) (FILE * __f, __warning_vector_t & __warnings)
- void [__gnu_profile::__trace_hashtable_size_free](#) ()
- void [__gnu_profile::__trace_hashtable_size_init](#) ()
- void [__gnu_profile::__trace_hashtable_size_report](#) (FILE * __f, __warning_vector_t & __warnings)
- void [__gnu_profile::__trace_list_to_slist_free](#) ()
- void [__gnu_profile::__trace_list_to_slist_init](#) ()
- void [__gnu_profile::__trace_list_to_slist_report](#) (FILE * __f, __warning_vector_t & __warnings)

- void **__gnu_profile::__trace_list_to_vector_free** ()
- void **__gnu_profile::__trace_list_to_vector_init** ()
- void **__gnu_profile::__trace_list_to_vector_report** (FILE * __f, __warning_vector_t & __warnings)
- void **__gnu_profile::__trace_map_to_unordered_map_free** ()
- void **__gnu_profile::__trace_map_to_unordered_map_init** ()
- void **__gnu_profile::__trace_map_to_unordered_map_report** (FILE * __f, __warning_vector_t & __warnings)
- template<typename __object_info, typename __stack_info >
void **__gnu_profile::__trace_report** (__trace_base< __object_info, __stack_info > * __cont, FILE * __f, __warning_vector_t & __warnings)
- void **__gnu_profile::__trace_vector_size_free** ()
- void **__gnu_profile::__trace_vector_size_init** ()
- void **__gnu_profile::__trace_vector_size_report** (FILE *, __warning_vector_t &)
- void **__gnu_profile::__trace_vector_to_list_free** ()
- void **__gnu_profile::__trace_vector_to_list_init** ()
- void **__gnu_profile::__trace_vector_to_list_report** (FILE *, __warning_vector_t &)
- void **__gnu_profile::__write_cost_factors** ()
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__trace_hash_func *, __S_hash_func, 0)
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__trace_hashtable_size *, __S_hashtable_size, 0)
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__trace_map2umap *, __S_map2umap, 0)
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__trace_vector_size *, __S_vector_size, 0)
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__trace_vector_to_list *, __S_vector_to_list, 0)
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__trace_list_to_slist *, __S_list_to_slist, 0)
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__trace_list_to_vector *, __S_list_to_vector, 0)
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__cost_factor, __vector_shift_cost_factor, {"__vector_↵
shift_cost_factor", 1.0})
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__cost_factor, __vector_iterate_cost_factor, {"__↵
vector_iterate_cost_factor", 1.0})
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__cost_factor, __vector_resize_cost_factor, {"__↵
vector_resize_cost_factor", 1.0})
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__cost_factor, __list_shift_cost_factor, {"__list_shift_↵
cost_factor", 0.0})
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__cost_factor, __list_iterate_cost_factor, {"__list_↵
iterate_cost_factor", 10.0})
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__cost_factor, __list_resize_cost_factor, {"__list_↵
resize_cost_factor", 0.0})
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__cost_factor, __map_insert_cost_factor, {"__map_↵
insert_cost_factor", 1.5})
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__cost_factor, __map_erase_cost_factor, {"__map_↵
erase_cost_factor", 1.5})
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__cost_factor, __map_find_cost_factor, {"__map_find_↵
cost_factor", 1})
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__cost_factor, __map_iterate_cost_factor, {"__map_↵
iterate_cost_factor", 2.3})
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__cost_factor, __umap_insert_cost_factor, {"__umap_↵
insert_cost_factor", 12.0})
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__cost_factor, __umap_erase_cost_factor, {"__umap_↵
erase_cost_factor", 12.0})
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__cost_factor, __umap_find_cost_factor, {"__umap_↵
find_cost_factor", 10.0})
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__cost_factor, __umap_iterate_cost_factor, {"__umap_↵
iterate_cost_factor", 1.7})
- **__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA** (__cost_factor_vector *, __cost_factors, 0)

- [__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA](#) (const char *, _S_trace_file_name, _GLIBCXX_PROFILE_TRACE_PATH_ROOT)
- [__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA](#) (std::size_t, _S_max_warn_count, _GLIBCXX_PROFILE_MAX_WARN_COUNT)
- [__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA](#) (std::size_t, _S_max_stack_depth, _GLIBCXX_PROFILE_MAX_STACK_DEPTH)
- [__gnu_profile::__GLIBCXX_PROFILE_DEFINE_DATA](#) (std::size_t, _S_max_mem, _GLIBCXX_PROFILE_MEMORY_PER_DIAGNOSTIC)
- [__gnu_profile::__GLIBCXX_PROFILE_DEFINE_UNINIT_DATA](#) (__env_t, __env)
- [__gnu_profile::__GLIBCXX_PROFILE_DEFINE_UNINIT_DATA](#) (__gnu_cxx::__mutex, __global_mutex)

6.448.1 Detailed Description

Data structures to represent profiling traces.

6.449 profiler_vector_size.h File Reference

Classes

- class [__gnu_profile::__trace_vector_size](#)

Namespaces

- [__gnu_profile](#)

Functions

- [__container_size_info * __gnu_profile::__trace_vector_size_construct](#) (std::size_t)
- [void __gnu_profile::__trace_vector_size_destruct](#) (__container_size_info *, std::size_t, std::size_t)
- [void __gnu_profile::__trace_vector_size_free](#) ()
- [void __gnu_profile::__trace_vector_size_init](#) ()
- [void __gnu_profile::__trace_vector_size_report](#) (FILE *, __warning_vector_t &)
- [void __gnu_profile::__trace_vector_size_resize](#) (__container_size_info *, std::size_t, std::size_t)

6.449.1 Detailed Description

Collection of vector size traces.

6.450 profiler_vector_to_list.h File Reference

Classes

- class [__gnu_profile::__trace_vector_to_list](#)
- class [__gnu_profile::__vector2list_info](#)
- class [__gnu_profile::__vector2list_stack_info](#)

Namespaces

- [__gnu_profile](#)

Functions

- `__vector2list_info * __gnu_profile::__trace_vector_to_list_construct ()`
- `void __gnu_profile::__trace_vector_to_list_destruct (__vector2list_info *)`
- `void __gnu_profile::__trace_vector_to_list_free ()`
- `void __gnu_profile::__trace_vector_to_list_init ()`
- `void __gnu_profile::__trace_vector_to_list_insert (__vector2list_info *, std::size_t, std::size_t)`
- `void __gnu_profile::__trace_vector_to_list_invalid_operator (__vector2list_info *)`
- `void __gnu_profile::__trace_vector_to_list_iterate (__vector2list_info *, int)`
- `void __gnu_profile::__trace_vector_to_list_report (FILE *, __warning_vector_t &)`
- `void __gnu_profile::__trace_vector_to_list_resize (__vector2list_info *, std::size_t, std::size_t)`

6.450.1 Detailed Description

diagnostics for vector to list.

6.451 propagate_const File Reference

Classes

- class [std::experimental::fundamentals_v2::propagate_const<_Tp>](#)

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_PROPAGATE_CONST`

Functions

- `template<typename _Tp >`
`constexpr const _Tp & std::experimental::fundamentals_v2::get_underlying (const propagate_const< _Tp >`
`&__pt) noexcept`
- `template<typename _Tp >`
`constexpr _Tp & std::experimental::fundamentals_v2::get_underlying (propagate_const< _Tp > &__pt)`
`noexcept`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v2::operator!= (const propagate_const< _Tp > &__pt,`
`nullptr_t)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v2::operator!= (nullptr_t, const propagate_const< _Tp >`
`&__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator!= (const propagate_const< _Tp > &__pt, const`
`propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator!= (const propagate_const< _Tp > &__pt, const`
`_Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator!= (const _Tp &__t, const propagate_const< ↵`
`_Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator< (const propagate_const< _Tp > &__pt, const`
`propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator< (const propagate_const< _Tp > &__pt, const`
`_Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator< (const _Tp &__t, const propagate_const< ↵`
`_Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator<= (const propagate_const< _Tp > &__pt,`
`const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator<= (const propagate_const< _Tp > &__pt,`
`const _Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator<= (const _Tp &__t, const propagate_const<`
`_Up > &__pu)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v2::operator== (const propagate_const< _Tp > &__pt,`
`nullptr_t)`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v2::operator== (nullptr_t, const propagate_const< _Tp >`
`&__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator== (const propagate_const< _Tp > &__pt,`
`const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator== (const propagate_const< _Tp > &__pt,`
`const _Up &__u)`

- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator== (const _Tp &__t, const propagate_const<`
`_Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator> (const propagate_const< _Tp > &__pt, const`
`propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator> (const propagate_const< _Tp > &__pt, const`
`_Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator> (const _Tp &__t, const propagate_const< ↵`
`_Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator>= (const propagate_const< _Tp > &__pt,`
`const propagate_const< _Up > &__pu)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator>= (const propagate_const< _Tp > &__pt,`
`const _Up &__u)`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v2::operator>= (const _Tp &__t, const propagate_const<`
`_Up > &__pu)`
- `template<typename _Tp >`
`constexpr void std::experimental::fundamentals_v2::swap (propagate_const< _Tp > &__pt, propagate_↵`
`const< _Tp > &__pt2) noexcept(__is_nothrow_swappable< _Tp >::value)`

6.451.1 Detailed Description

This is a TS C++ Library header.

6.452 ptr_traits.h File Reference

Classes

- struct [std::pointer_traits<_Ptr>](#)
- struct [std::pointer_traits<_Tp*>](#)

Namespaces

- [std](#)

Typedefs

- `template<typename _Tp >`
`using std::__get_first_arg_t = typename __get_first_arg< _Tp >::type`
- `template<typename _Tp >`
`using std::__make_not_void = typename conditional< is_void< _Tp >::value, __undefined, _Tp >::type`
- `template<typename _Ptr, typename _Tp >`
`using std::__ptr_rebind = typename pointer_traits< _Ptr >::template rebind< _Tp >`
- `template<typename _Tp, typename _Up >`
`using std::__replace_first_arg_t = typename __replace_first_arg< _Tp, _Up >::type`

6.452.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

6.453 quadratic_probe_fn_imp.hpp File Reference

6.453.1 Detailed Description

Contains a probe policy implementation

6.454 queue File Reference

Macros

- `#define _GLIBCXX_QUEUE`

6.454.1 Detailed Description

This is a Standard C++ Library header.

6.455 queue.h File Reference

Classes

- class [__gnu_parallel::_RestrictedBoundedConcurrentQueue<_Tp>](#)

Namespaces

- [__gnu_parallel](#)

Macros

- `#define _GLIBCXX_VOLATILE`

6.455.1 Detailed Description

Lock-free double-ended queue. This file is a GNU parallel extension to the Standard C++ Library.

6.455.2 Macro Definition Documentation

6.455.2.1 `#define _GLIBCXX_VOLATILE`

Decide whether to declare certain variable volatile in this file.

Definition at line 40 of file queue.h.

6.456 `quicksort.h` File Reference

Namespaces

- [__gnu_parallel](#)

Functions

- `template<typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_sort_qs (_RAIter __begin, _RAIter __end, _Compare __comp, _ThreadIndex __num_threads)`
- `template<typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_sort_qs_conquer (_RAIter __begin, _RAIter __end, _Compare __comp, _ThreadIndex __num_threads)`
- `template<typename _RAIter, typename _Compare >`
`std::iterator_traits< _RAIter >::difference_type __gnu_parallel::__parallel_sort_qs_divide (_RAIter __begin, _RAIter __end, _Compare __comp, typename std::iterator_traits< _RAIter >::difference_type __pivot_rank, typename std::iterator_traits< _RAIter >::difference_type __num_samples, _ThreadIndex __num_threads)`

6.456.1 Detailed Description

Implementation of a unbalanced parallel quicksort (in-place). This file is a GNU parallel extension to the Standard C++ Library.

6.457 `quoted_string.h` File Reference

Classes

- `struct std::__detail::__Quoted_string< _String, _CharT >`

Namespaces

- [std](#)
- [std::__detail](#)

Functions

- `template<typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::__detail::operator<< (std::basic_ostream< _CharT, _Traits > &↵`
`__os, const _Quoted_string< const _CharT *, _CharT > &__str)`
- `template<typename _CharT, typename _Traits, typename _String >`
`std::basic_ostream< _CharT, _Traits > & std::__detail::operator<< (std::basic_ostream< _CharT, _Traits > &↵`
`__os, const _Quoted_string< _String, _CharT > &__str)`
- `template<typename _CharT, typename _Traits, typename _Alloc >`
`std::basic_istream< _CharT, _Traits > & std::__detail::operator>> (std::basic_istream< _CharT, _Traits > &↵`
`_is, const _Quoted_string< basic_string< _CharT, _Traits, _Alloc > &, _CharT > &__str)`

6.457.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iomanip>`.

6.458 r_erase_fn_imps.hpp File Reference

6.458.1 Detailed Description

Contains an implementation class for `bin_search_tree_`.

6.459 r_erase_fn_imps.hpp File Reference

6.459.1 Detailed Description

Contains an implementation class for `pat_trie`.

6.460 random File Reference

Macros

- `#define _GLIBCXX_RANDOM`

6.460.1 Detailed Description

This is a Standard C++ Library header.

6.461 random File Reference

Namespaces

- `std`

Macros

- `#define __cpp_lib_experimental_randint`
- `#define _GLIBCXX_EXPERIMENTAL_RANDOM`

Functions

- `std::default_random_engine & std::experimental::fundamentals_v2::S_randint_engine ()`
- `template<typename _IntType >
_IntType std::experimental::fundamentals_v2::randint (_IntType __a, _IntType __b)`
- `void std::experimental::fundamentals_v2::reseed ()`
- `void std::experimental::fundamentals_v2::reseed (default_random_engine::result_type __value)`

6.461.1 Detailed Description

This is a TS C++ Library header.

6.462 random.h File Reference

Classes

- class `std::bernoulli_distribution`
- struct `std::bernoulli_distribution::param_type`
- class `std::binomial_distribution< _IntType >`
- struct `std::binomial_distribution< _IntType >::param_type`
- class `std::cauchy_distribution< _RealType >`
- struct `std::cauchy_distribution< _RealType >::param_type`
- class `std::chi_squared_distribution< _RealType >`
- struct `std::chi_squared_distribution< _RealType >::param_type`
- class `std::discard_block_engine< _RandomNumberEngine, __p, __r >`
- class `std::discrete_distribution< _IntType >`
- struct `std::discrete_distribution< _IntType >::param_type`
- class `std::exponential_distribution< _RealType >`
- struct `std::exponential_distribution< _RealType >::param_type`
- class `std::extreme_value_distribution< _RealType >`
- struct `std::extreme_value_distribution< _RealType >::param_type`
- class `std::fisher_f_distribution< _RealType >`
- struct `std::fisher_f_distribution< _RealType >::param_type`
- class `std::gamma_distribution< _RealType >`
- struct `std::gamma_distribution< _RealType >::param_type`
- class `std::geometric_distribution< _IntType >`
- struct `std::geometric_distribution< _IntType >::param_type`
- class `std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >`
- class `std::linear_congruential_engine< _UIntType, __a, __c, __m >`
- class `std::lognormal_distribution< _RealType >`
- struct `std::lognormal_distribution< _RealType >::param_type`

- class [std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >](#)
- class [std::negative_binomial_distribution< _IntType >](#)
- struct [std::negative_binomial_distribution< _IntType >::param_type](#)
- class [std::normal_distribution< _RealType >](#)
- struct [std::normal_distribution< _RealType >::param_type](#)
- class [std::piecewise_constant_distribution< _RealType >](#)
- struct [std::piecewise_constant_distribution< _RealType >::param_type](#)
- class [std::piecewise_linear_distribution< _RealType >](#)
- struct [std::piecewise_linear_distribution< _RealType >::param_type](#)
- class [std::poisson_distribution< _IntType >](#)
- struct [std::poisson_distribution< _IntType >::param_type](#)
- class [std::random_device](#)
- class [std::seed_seq](#)
- class [std::shuffle_order_engine< _RandomNumberEngine, __k >](#)
- class [std::student_t_distribution< _RealType >](#)
- struct [std::student_t_distribution< _RealType >::param_type](#)
- class [std::subtract_with_carry_engine< _UIntType, __w, __s, __r >](#)
- class [std::uniform_real_distribution< _RealType >](#)
- struct [std::uniform_real_distribution< _RealType >::param_type](#)
- class [std::weibull_distribution< _RealType >](#)
- struct [std::weibull_distribution< _RealType >::param_type](#)

Namespaces

- [std](#)
- [std::__detail](#)

Typedefs

- typedef minstd_rand0 **std::default_random_engine**
- typedef shuffle_order_engine< minstd_rand0, 256 > **std::knuth_b**
- typedef linear_congruential_engine< uint_fast32_t, 48271UL, 0UL, 2147483647UL > [std::minstd_rand](#)
- typedef linear_congruential_engine< uint_fast32_t, 16807UL, 0UL, 2147483647UL > [std::minstd_rand0](#)
- typedef mersenne_twister_engine< uint_fast32_t, 32, 624, 397, 31, 0x9908b0dfUL, 11, 0xffffffffUL, 7, 0x9d2c5680UL, 15, 0xefc60000UL, 18, 1812433253UL > [std::mt19937](#)
- typedef mersenne_twister_engine< uint_fast64_t, 64, 312, 156, 31, 0xb5026f5aa96619e9ULL, 29, 0x5555555555555555ULL, 17, 0x71d67ffeda60000ULL, 37, 0xffff7eee00000000ULL, 43, 6364136223846793005ULL > [std::mt19937_64](#)
- typedef discard_block_engine< ranlux24_base, 223, 23 > **std::ranlux24**
- typedef subtract_with_carry_engine< uint_fast32_t, 24, 10, 24 > **std::ranlux24_base**
- typedef discard_block_engine< ranlux48_base, 389, 11 > **std::ranlux48**
- typedef subtract_with_carry_engine< uint_fast64_t, 48, 5, 12 > **std::ranlux48_base**

Functions

- `template<typename _RealType, size_t __bits, typename _UniformRandomNumberGenerator >
_RealType std::generate_canonical (_UniformRandomNumberGenerator &__g)`
- `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m>
bool std::operator!= (const std::linear_congruential_engine< _UIntType, __a, __c, __m > &__lhs, const std::linear_congruential_engine< _UIntType, __a, __c, __m > &__rhs)`
- `template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _UIntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f>
bool std::operator!= (const std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f > &__lhs, const std::mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f > &__rhs)`
- `template<typename _UIntType, size_t __w, size_t __s, size_t __r>
bool std::operator!= (const std::subtract_with_carry_engine< _UIntType, __w, __s, __r > &__lhs, const std::subtract_with_carry_engine< _UIntType, __w, __s, __r > &__rhs)`
- `template<typename _RandomNumberEngine, size_t __p, size_t __r>
bool std::operator!= (const std::discard_block_engine< _RandomNumberEngine, __p, __r > &__lhs, const std::discard_block_engine< _RandomNumberEngine, __p, __r > &__rhs)`
- `template<typename _RandomNumberEngine, size_t __w, typename _UIntType >
bool std::operator!= (const std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > &__lhs, const std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > &__rhs)`
- `template<typename _RandomNumberEngine, size_t __k>
bool std::operator!= (const std::shuffle_order_engine< _RandomNumberEngine, __k > &__lhs, const std::shuffle_order_engine< _RandomNumberEngine, __k > &__rhs)`
- `template<typename _IntType >
bool std::operator!= (const std::uniform_int_distribution< _IntType > &__d1, const std::uniform_int_distribution< _IntType > &__d2)`
- `template<typename _IntType >
bool std::operator!= (const std::uniform_real_distribution< _IntType > &__d1, const std::uniform_real_distribution< _IntType > &__d2)`
- `template<typename _RealType >
bool std::operator!= (const std::normal_distribution< _RealType > &__d1, const std::normal_distribution< _RealType > &__d2)`
- `template<typename _RealType >
bool std::operator!= (const std::lognormal_distribution< _RealType > &__d1, const std::lognormal_distribution< _RealType > &__d2)`
- `template<typename _RealType >
bool std::operator!= (const std::gamma_distribution< _RealType > &__d1, const std::gamma_distribution< _RealType > &__d2)`
- `template<typename _RealType >
bool std::operator!= (const std::chi_squared_distribution< _RealType > &__d1, const std::chi_squared_distribution< _RealType > &__d2)`
- `template<typename _RealType >
bool std::operator!= (const std::cauchy_distribution< _RealType > &__d1, const std::cauchy_distribution< _RealType > &__d2)`
- `template<typename _RealType >
bool std::operator!= (const std::fisher_f_distribution< _RealType > &__d1, const std::fisher_f_distribution< _RealType > &__d2)`
- `template<typename _RealType >
bool std::operator!= (const std::student_t_distribution< _RealType > &__d1, const std::student_t_distribution< _RealType > &__d2)`
- `bool std::operator!= (const std::bernoulli_distribution &__d1, const std::bernoulli_distribution &__d2)`

- `template<typename _IntType >`
`bool std::operator!= (const std::binomial_distribution< _IntType > &__d1, const std::binomial_distribution< _IntType > &__d2)`
- `template<typename _IntType >`
`bool std::operator!= (const std::geometric_distribution< _IntType > &__d1, const std::geometric_distribution< _IntType > &__d2)`
- `template<typename _IntType >`
`bool std::operator!= (const std::negative_binomial_distribution< _IntType > &__d1, const std::negative_binomial_distribution< _IntType > &__d2)`
- `template<typename _IntType >`
`bool std::operator!= (const std::poisson_distribution< _IntType > &__d1, const std::poisson_distribution< _IntType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!= (const std::exponential_distribution< _RealType > &__d1, const std::exponential_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!= (const std::weibull_distribution< _RealType > &__d1, const std::weibull_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!= (const std::extreme_value_distribution< _RealType > &__d1, const std::extreme_value_distribution< _RealType > &__d2)`
- `template<typename _IntType >`
`bool std::operator!= (const std::discrete_distribution< _IntType > &__d1, const std::discrete_distribution< _IntType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!= (const std::piecewise_constant_distribution< _RealType > &__d1, const std::piecewise_constant_distribution< _RealType > &__d2)`
- `template<typename _RealType >`
`bool std::operator!= (const std::piecewise_linear_distribution< _RealType > &__d1, const std::piecewise_linear_distribution< _RealType > &__d2)`
- `template<typename _RandomNumberEngine, size_t __w, typename _UIntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &, const std::uniform_int_distribution< _IntType > &)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &, const std::uniform_real_distribution< _RealType > &)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const std::cauchy_distribution< _RealType > &__x)`
- `template<typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const std::bernoulli_distribution &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const std::geometric_distribution< _IntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const std::exponential_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const std::weibull_distribution< _RealType > &__x)`

- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const
std::extreme_value_distribution< _RealType > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &, std::←
::uniform_int_distribution< _IntType > &)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &, std::←
::uniform_real_distribution< _RealType > &)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is, std::←
::cauchy_distribution< _RealType > &__x)`
- `template<typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is, std::←
::bernoulli_distribution &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is, std::←
::geometric_distribution< _IntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is, std::←
::exponential_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is, std::←
::weibull_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is, std::←
::extreme_value_distribution< _RealType > &__x)`

6.462.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<random>`.

6.463 random.tcc File Reference

Namespaces

- `std`
- `std::__detail`

Macros

- `#define _RANDOM_TCC`

Functions

- `template<typename _InputIterator, typename _OutputIterator, typename _Tp >
_OutputIterator std::detail::__normalize (_InputIterator __first, _InputIterator __last, _OutputIterator __result,
const _Tp &__factor)`
- `template<typename _RealType, size_t __bits, typename _UniformRandomNumberGenerator >
_RealType std::generate_canonical (_UniformRandomNumberGenerator &__g)`
- `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,
const linear_congruential_engine< _UIntType, __a, __c, __m > &__lcr)`
- `template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _U↵
IntType __b, size_t __t, _UIntType __c, size_t __l, _UIntType __f, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,
const mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f
> &__x)`
- `template<typename _UIntType, size_t __w, size_t __s, size_t __r, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,
const subtract_with_carry_engine< _UIntType, __w, __s, __r > &__x)`
- `template<typename _RandomNumberEngine, size_t __p, size_t __r, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,
const discard_block_engine< _RandomNumberEngine, __p, __r > &__x)`
- `template<typename _RandomNumberEngine, size_t __k, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,
const shuffle_order_engine< _RandomNumberEngine, __k > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,
const negative_binomial_distribution< _IntType > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,
const poisson_distribution< _IntType > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,
const binomial_distribution< _IntType > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__, const std::↵
::uniform_int_distribution< _IntType > &)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__, const std::↵
::uniform_real_distribution< _RealType > &)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,
const normal_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,
const lognormal_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,
const chi_squared_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,
const fisher_f_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >
std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,
const student_t_distribution< _RealType > &__x)`

- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,`
`const gamma_distribution< _RealType > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,`
`const discrete_distribution< _IntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,`
`const piecewise_constant_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::cauchy_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os,`
`const piecewise_linear_distribution< _RealType > &__x)`
- `template<typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::bernoulli_distribution &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::geometric_distribution< _IntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::exponential_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::weibull_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & std::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const`
`std::extreme_value_distribution< _RealType > &__x)`
- `template<typename _RealType >`
`bool std::operator== (const std::normal_distribution< _RealType > &__d1, const std::normal_distribution< _RealType > &__d2)`
- `template<typename _UIntType, _UIntType __a, _UIntType __c, _UIntType __m, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`linear_congruential_engine< _UIntType, __a, __c, __m > &__lcr)`
- `template<typename _UIntType, size_t __w, size_t __n, size_t __m, size_t __r, _UIntType __a, size_t __u, _UIntType __d, size_t __s, _UIntType __b,`
`size_t __t, _UIntType __c, size_t __l, _UIntType __f, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`mersenne_twister_engine< _UIntType, __w, __n, __m, __r, __a, __u, __d, __s, __b, __t, __c, __l, __f >`
`&__x)`
- `template<typename _UIntType, size_t __w, size_t __s, size_t __r, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`subtract_with_carry_engine< _UIntType, __w, __s, __r > &__x)`
- `template<typename _RandomNumberEngine, size_t __p, size_t __r, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`discard_block_engine< _RandomNumberEngine, __p, __r > &__x)`
- `template<typename _RandomNumberEngine, size_t __k, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`shuffle_order_engine< _RandomNumberEngine, __k > &__x)`
- `template<typename _IntType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & std::operator>> (std::basic_istream< _CharT, _Traits > &__is,`
`negative_binomial_distribution< _IntType > &__x)`

- [illegible]

6.463.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<random>`.

6.464 random.tcc File Reference

Namespaces

- [__gnu_cxx](#)

Macros

- `#define _EXT_RANDOM_TCC`

Functions

- `template<typename _UIntType, size_t __m, size_t __pos1, size_t __sl1, size_t __sl2, size_t __sr1, size_t __sr2, uint32_t __msk1, uint32_t __msk2, uint32_t __msk3, uint32_t __msk4, uint32_t __parity1, uint32_t __parity2, uint32_t __parity3, uint32_t __parity4, typename _CharT, typename _Traits>`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > & __os, const __gnu_cxx::simd_fast_mersenne_twister_engine< _UIntType, __m, __pos1, __sl1, __sl2, __sr1, __sr2, __msk1, __msk2, __msk3, __msk4, __parity1, __parity2, __parity3, __parity4 > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits>`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > & __os, const __gnu_cxx::beta_distribution< _RealType > & __x)`
- `template<size_t _Dimen, typename _RealType, typename _CharT, typename _Traits>`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > & __os, const __gnu_cxx::normal_mv_distribution< _Dimen, _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits>`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > & __os, const rice_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits>`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > & __os, const nakagami_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits>`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > & __os, const pareto_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits>`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > & __os, const k_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits>`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > & __os, const arcsine_distribution< _RealType > & __x)`
- `template<typename _RealType, typename _CharT, typename _Traits>`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > & __os, const hoyt_distribution< _RealType > & __x)`

- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const __gnu_cxx::triangular_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const __gnu_cxx::von_mises_distribution< _RealType > &__x)`
- `template<typename _UIntType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const __gnu_cxx::hypergeometric_distribution< _UIntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const logistic_distribution< _RealType > &__x)`
- `template<std::size_t _Dimen, typename _RealType, typename _CharT, typename _Traits >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &__os, const __gnu_cxx::uniform_on_sphere_distribution< _Dimen, _RealType > &__x)`
- `template<typename _UIntType, size_t __m, size_t __pos1, size_t __sl1, size_t __sl2, size_t __sr1, size_t __sr2, uint32_t __msk1, uint32_t __msk2, uint32_t __msk3, uint32_t __msk4, uint32_t __parity1, uint32_t __parity2, uint32_t __parity3, uint32_t __parity4>`
`bool __gnu_cxx::operator== (const __gnu_cxx::simd_fast_mersenne_twister_engine< _UIntType, __m, __pos1, __sl1, __sl2, __sr1, __sr2, __msk1, __msk2, __msk3, __msk4, __parity1, __parity2, __parity3, __parity4 > &__lhs, const __gnu_cxx::simd_fast_mersenne_twister_engine< _UIntType, __m, __pos1, __sl1, __sl2, __sr1, __sr2, __msk1, __msk2, __msk3, __msk4, __parity1, __parity2, __parity3, __parity4 > &__rhs)`
- `template<size_t _Dimen, typename _RealType >`
`bool __gnu_cxx::operator== (const __gnu_cxx::normal_mv_distribution< _Dimen, _RealType > &__d1, const __gnu_cxx::normal_mv_distribution< _Dimen, _RealType > &__d2)`
- `template<typename _UIntType, size_t __m, size_t __pos1, size_t __sl1, size_t __sl2, size_t __sr1, size_t __sr2, uint32_t __msk1, uint32_t __msk2, uint32_t __msk3, uint32_t __msk4, uint32_t __parity1, uint32_t __parity2, uint32_t __parity3, uint32_t __parity4, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & __gnu_cxx::operator>> (std::basic_istream< _CharT, _Traits > &__is, __gnu_cxx::simd_fast_mersenne_twister_engine< _UIntType, __m, __pos1, __sl1, __sl2, __sr1, __sr2, __msk1, __msk2, __msk3, __msk4, __parity1, __parity2, __parity3, __parity4 > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & __gnu_cxx::operator>> (std::basic_istream< _CharT, _Traits > &__is, __gnu_cxx::beta_distribution< _RealType > &__x)`
- `template<size_t _Dimen, typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & __gnu_cxx::operator>> (std::basic_istream< _CharT, _Traits > &__is, __gnu_cxx::normal_mv_distribution< _Dimen, _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & __gnu_cxx::operator>> (std::basic_istream< _CharT, _Traits > &__is, rice_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & __gnu_cxx::operator>> (std::basic_istream< _CharT, _Traits > &__is, nakagami_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & __gnu_cxx::operator>> (std::basic_istream< _CharT, _Traits > &__is, pareto_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & __gnu_cxx::operator>> (std::basic_istream< _CharT, _Traits > &__is, k_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & __gnu_cxx::operator>> (std::basic_istream< _CharT, _Traits > &__is, arcsine_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & __gnu_cxx::operator>> (std::basic_istream< _CharT, _Traits > &__is, hoyt_distribution< _RealType > &__x)`

- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & __gnu_cxx::operator>> (std::basic_istream< _CharT, _Traits > &←`
`__is, __gnu_cxx::triangular_distribution< _RealType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & __gnu_cxx::operator>> (std::basic_istream< _CharT, _Traits > &←`
`__is, __gnu_cxx::von_mises_distribution< _RealType > &__x)`
- `template<typename _UIntType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & __gnu_cxx::operator>> (std::basic_istream< _CharT, _Traits > &←`
`__is, __gnu_cxx::hypergeometric_distribution< _UIntType > &__x)`
- `template<typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & __gnu_cxx::operator>> (std::basic_istream< _CharT, _Traits > &←`
`__is, logistic_distribution< _RealType > &__x)`
- `template<std::size_t _Dimen, typename _RealType, typename _CharT, typename _Traits >`
`std::basic_istream< _CharT, _Traits > & __gnu_cxx::operator>> (std::basic_istream< _CharT, _Traits > &←`
`__is, __gnu_cxx::uniform_on_sphere_distribution< _Dimen, _RealType > &__x)`

6.464.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ext/random>`.

6.465 random_number.h File Reference

Classes

- class `__gnu_parallel::_RandomNumber`

Namespaces

- `__gnu_parallel`

6.465.1 Detailed Description

Random number generator based on the Mersenne twister. This file is a GNU parallel extension to the Standard C++ Library.

6.466 random_shuffle.h File Reference

Classes

- struct `__gnu_parallel::_DRandomShufflingGlobalData< _RAIter >`
- struct `__gnu_parallel::_DRSSorterPU< _RAIter, _RandomNumberGenerator >`

Namespaces

- [__gnu_parallel](#)

Typedefs

- typedef unsigned short [__gnu_parallel::BinIndex](#)

Functions

- template<typename _RAIter, typename _RandomNumberGenerator >
void [__gnu_parallel::parallel_random_shuffle](#) (_RAIter __begin, _RAIter __end, _RandomNumberGenerator __rng=_RandomNumber())
- template<typename _RAIter, typename _RandomNumberGenerator >
void [__gnu_parallel::parallel_random_shuffle_drs](#) (_RAIter __begin, _RAIter __end, typename std::iterator_traits<_RAIter>::difference_type __n, _ThreadIndex __num_threads, _RandomNumberGenerator &__rng)
- template<typename _RAIter, typename _RandomNumberGenerator >
void [__gnu_parallel::parallel_random_shuffle_drs_pu](#) (_DRSSorterPU<_RAIter, _RandomNumberGenerator > *__pus)
- template<typename _RandomNumberGenerator >
int [__gnu_parallel::random_number_pow2](#) (int __logp, _RandomNumberGenerator &__rng)
- template<typename _Tp >
_Tp [__gnu_parallel::round_up_to_pow2](#) (_Tp __x)
- template<typename _RAIter, typename _RandomNumberGenerator >
void [__gnu_parallel::sequential_random_shuffle](#) (_RAIter __begin, _RAIter __end, _RandomNumberGenerator &__rng)

6.466.1 Detailed Description

Parallel implementation of std::random_shuffle(). This file is a GNU parallel extension to the Standard C++ Library.

6.467 range_access.h File Reference

Classes

- class [std::valarray<_Tp>](#)

Namespaces

- [std](#)

Functions

- `template<typename _Container >`
`auto std::begin (_Container &__cont) -> decltype(__cont.begin())`
- `template<typename _Container >`
`auto std::begin (const _Container &__cont) -> decltype(__cont.begin())`
- `template<typename _Tp, size_t _Nm>`
`_GLIBCXX14_CONSTEXPR _Tp * std::begin (_Tp(&__arr)[_Nm])`
- `template<class _Tp >`
`_Tp * std::begin (valarray< _Tp > &__va)`
- `template<class _Tp >`
`const _Tp * std::begin (const valarray< _Tp > &__va)`
- `template<typename _Container >`
`constexpr auto std::cbegin (const _Container &__cont) noexcept(noexcept(std::begin(__cont))) -> decltype(std::begin(__cont))`
- `template<typename _Container >`
`constexpr auto std::cend (const _Container &__cont) noexcept(noexcept(std::end(__cont))) -> decltype(std::end(__cont))`
- `template<typename _Container >`
`auto std::rbegin (const _Container &__cont) -> decltype(std::rbegin(__cont))`
- `template<typename _Container >`
`auto std::rend (const _Container &__cont) -> decltype(std::rend(__cont))`
- `template<typename _Container >`
`auto std::end (_Container &__cont) -> decltype(__cont.end())`
- `template<typename _Container >`
`auto std::end (const _Container &__cont) -> decltype(__cont.end())`
- `template<typename _Tp, size_t _Nm>`
`_GLIBCXX14_CONSTEXPR _Tp * std::end (_Tp(&__arr)[_Nm])`
- `template<class _Tp >`
`_Tp * std::end (valarray< _Tp > &__va)`
- `template<class _Tp >`
`const _Tp * std::end (const valarray< _Tp > &__va)`
- `template<typename _Container >`
`auto std::rbegin (_Container &__cont) -> decltype(__cont.rbegin())`
- `template<typename _Container >`
`auto std::rbegin (const _Container &__cont) -> decltype(__cont.rbegin())`
- `template<typename _Tp, size_t _Nm>`
`reverse_iterator< _Tp * > std::rbegin (_Tp(&__arr)[_Nm])`
- `template<typename _Tp >`
`reverse_iterator< const _Tp * > std::rbegin (initializer_list< _Tp > __il)`
- `template<typename _Container >`
`auto std::rend (_Container &__cont) -> decltype(__cont.rend())`
- `template<typename _Container >`
`auto std::rend (const _Container &__cont) -> decltype(__cont.rend())`
- `template<typename _Tp, size_t _Nm>`
`reverse_iterator< _Tp * > std::rend (_Tp(&__arr)[_Nm])`
- `template<typename _Tp >`
`reverse_iterator< const _Tp * > std::rend (initializer_list< _Tp > __il)`

6.467.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iterator>`.

6.468 ranged_hash_fn.hpp File Reference

Classes

- class [__gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, Store_Hash >](#)
- class [__gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, false >](#)
- class [__gnu_pbds::detail::ranged_hash_fn< Key, Hash_Fn, _Alloc, Comb_Hash_Fn, true >](#)
- class [__gnu_pbds::detail::ranged_hash_fn< Key, null_type, _Alloc, Comb_Hash_Fn, false >](#)
- class [__gnu_pbds::detail::ranged_hash_fn< Key, null_type, _Alloc, Comb_Hash_Fn, true >](#)

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`

6.468.1 Detailed Description

Contains a unified ranged hash functor, allowing the hash tables to deal with a single class for ranged hashing.

6.469 ranged_probe_fn.hpp File Reference

Classes

- class [__gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, Store_Hash >](#)
- class [__gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, false >](#)
- class [__gnu_pbds::detail::ranged_probe_fn< Key, Hash_Fn, _Alloc, Comb_Probe_Fn, Probe_Fn, true >](#)
- class [__gnu_pbds::detail::ranged_probe_fn< Key, null_type, _Alloc, Comb_Probe_Fn, null_type, false >](#)

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`

6.469.1 Detailed Description

Contains a unified ranged probe functor, allowing the probe tables to deal with a single class for ranged probing.

6.470 ratio File Reference

Classes

- struct [std::ratio<_Num, _Den >](#)
- struct [std::ratio_equal<_R1, _R2 >](#)
- struct [std::ratio_not_equal<_R1, _R2 >](#)

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_RATIO`

Typedefs

- `template<typename _R1, typename _R2 >`
`using std::ratio_divide = typename __ratio_divide<_R1, _R2 >::type`
- `template<typename _R1, typename _R2 >`
`using std::ratio_multiply = typename __ratio_multiply<_R1, _R2 >::type`

6.470.1 Detailed Description

This is a Standard C++ Library header.

6.471 ratio File Reference

Namespaces

- [std](#)
- [std::tr2](#)

6.471.1 Detailed Description

This is a TR2 C++ Library header.

6.472 ratio File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_RATIO`

Variables

- `template<typename _R1, typename _R2 >`
`constexpr bool std::experimental::fundamentals_v1::ratio_equal_v`
- `template<typename _R1, typename _R2 >`
`constexpr bool std::experimental::fundamentals_v1::ratio_greater_equal_v`
- `template<typename _R1, typename _R2 >`
`constexpr bool std::experimental::fundamentals_v1::ratio_greater_v`
- `template<typename _R1, typename _R2 >`
`constexpr bool std::experimental::fundamentals_v1::ratio_less_equal_v`
- `template<typename _R1, typename _R2 >`
`constexpr bool std::experimental::fundamentals_v1::ratio_less_v`
- `template<typename _R1, typename _R2 >`
`constexpr bool std::experimental::fundamentals_v1::ratio_not_equal_v`

6.472.1 Detailed Description

This is a TS C++ Library header.

6.473 rb_tree File Reference

Classes

- `struct __gnu_cxx::rb_tree< _Key, _Value, _KeyOfValue, _Compare, _Alloc >`

Namespaces

- [__gnu_cxx](#)

Macros

- `#define _RB_TREE`

6.473.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGI STL subset).

6.474 `rb_tree.hpp` File Reference

Classes

- class [`__gnu_pbds::detail::rb_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >`](#)

Namespaces

- [`__gnu_pbds`](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_RB_TREE_BASE`
- `#define PB_DS_RB_TREE_BASE_NAME`
- `#define PB_DS_RB_TREE_NAME`
- `#define PB_DS_STRUCT_ONLY_ASSERT_VALID(X)`

6.474.1 Detailed Description

Contains an implementation for Red Black trees.

6.475 `rc.hpp` File Reference

Classes

- class [`__gnu_pbds::detail::rc< _Node, _Alloc >`](#)

Namespaces

- [`__gnu_pbds`](#)

6.475.1 Detailed Description

Contains a redundant (binary counter).

6.476 rc_binomial_heap_.hpp File Reference

Classes

- class [__gnu_pbds::detail::rc_binomial_heap< Value_Type, Cmp_Fn, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_RC_C_DEC`

6.476.1 Detailed Description

Contains an implementation for redundant-counter binomial heap.

6.477 rc_string_base.h File Reference

Classes

- class [__gnu_cxx::__rc_string_base< _CharT, _Traits, _Alloc >](#)

Namespaces

- [__gnu_cxx](#)

6.477.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ext/vstring.h>`.

6.478 regex File Reference

Macros

- `#define _GLIBCXX_REGEX`

6.478.1 Detailed Description

This is a Standard C++ Library header.

6.479 regex File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_REGEX`

Typedefs

- `typedef match_results< const char * > std::experimental::fundamentals_v2::pmr::cmatch`
- `template<typename _BidirectionalIterator >`
`using std::experimental::fundamentals_v2::pmr::match_results = std::match_results< _BidirectionalIterator,`
`polymorphic_allocator< sub_match< _BidirectionalIterator >>>`
- `typedef match_results< string::const_iterator > std::experimental::fundamentals_v2::pmr::smatch`
- `typedef match_results< const wchar_t * > std::experimental::fundamentals_v2::pmr::wcmatch`
- `typedef match_results< wstring::const_iterator > std::experimental::fundamentals_v2::pmr::wsmatch`

6.479.1 Detailed Description

This is a TS C++ Library header.

6.480 regex.h File Reference

Classes

- `class std::__detail::_Executor< _Bilter, _Alloc, _TraitsT, __dfs_mode >`
- `class std::basic_regex< _Ch_type, _Rx_traits >`
- `class std::basic_regex< _Ch_type, _Rx_traits >`
- `class std::match_results< _Bi_iter, _Alloc >`
- `class std::match_results< _Bi_iter, _Alloc >`
- `class std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >`
- `class std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >`
- `struct std::regex_traits< _Ch_type >`
- `class std::sub_match< _Bilter >`

Namespaces

- [std](#)
- [std::__detail](#)

Typedefs

- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
using **std::__sub_match_string** = `basic_string< typename iterator_traits< _Bi_iter >::value_type, _Ch_traits, _Ch_alloc >`
- `typedef match_results< const char * >` **std::cmatch**
- `typedef regex_iterator< const char * >` **std::cregex_iterator**
- `typedef regex_token_iterator< const char * >` [std::cregex_token_iterator](#)
- `typedef sub_match< const char * >` [std::csub_match](#)
- `typedef basic_regex< char >` [std::regex](#)
- `typedef match_results< string::const_iterator >` **std::smatch**
- `typedef regex_iterator< string::const_iterator >` **std::sregex_iterator**
- `typedef regex_token_iterator< string::const_iterator >` [std::sregex_token_iterator](#)
- `typedef sub_match< string::const_iterator >` [std::ssub_match](#)
- `typedef match_results< const wchar_t * >` **std::wcmatch**
- `typedef regex_iterator< const wchar_t * >` **std::wcregex_iterator**
- `typedef regex_token_iterator< const wchar_t * >` [std::wcregex_token_iterator](#)
- `typedef sub_match< const wchar_t * >` [std::wcsb_match](#)
- `typedef basic_regex< wchar_t >` [std::wregex](#)
- `typedef match_results< wstring::const_iterator >` **std::wsmatch**
- `typedef regex_iterator< wstring::const_iterator >` **std::wsregex_iterator**
- `typedef regex_token_iterator< wstring::const_iterator >` [std::wsregex_token_iterator](#)
- `typedef sub_match< wstring::const_iterator >` [std::wssub_match](#)

Enumerations

- `enum _RegexExecutorPolicy : int { _S_auto, _S_alternate }`

Functions

- `template<typename _Bilter, typename _Alloc, typename _CharT, typename _TraitsT, _RegexExecutorPolicy __policy, bool __match_↔ mode>`
`bool std::__detail::__regex_algo_impl (_Bilter __s, _Bilter __e, match_results< _Bilter, _Alloc > &__m, const basic_regex< _CharT, _TraitsT > &__re, regex_constants::match_flag_type __flags)`
- `template<typename _Bilter >`
`bool std::operator!= (const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)`
- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
`bool std::operator!= (const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
`bool std::operator!= (const sub_match< _Bi_iter > &__lhs, const __sub_match_string< _Bi_iter, _Ch_traits, _↔ Ch_alloc > &__rhs)`

- `template<typename _Bi_iter >`
`bool std::operator!= (typename iterator_traits< _Bi_iter >::value_type const *__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator!= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const *__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator!= (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator!= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Bi_iter, class _Alloc >`
`bool std::operator!= (const match_results< _Bi_iter, _Alloc > &__m1, const match_results< _Bi_iter, _Alloc > &__m2)`
- `template<typename _Bilter >`
`bool std::operator< (const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)`
- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
`bool std::operator< (const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter, class _Ch_traits, class _Ch_alloc >`
`bool std::operator< (const sub_match< _Bi_iter > &__lhs, const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator< (typename iterator_traits< _Bi_iter >::value_type const *__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator< (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const *__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator< (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator< (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Ch_type, typename _Ch_traits, typename _Bi_iter >`
`basic_ostream< _Ch_type, _Ch_traits > & std::operator<< (basic_ostream< _Ch_type, _Ch_traits > &__os, const sub_match< _Bi_iter > &__m)`
- `template<typename _Bilter >`
`bool std::operator<= (const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)`
- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
`bool std::operator<= (const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter, class _Ch_traits, class _Ch_alloc >`
`bool std::operator<= (const sub_match< _Bi_iter > &__lhs, const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator<= (typename iterator_traits< _Bi_iter >::value_type const *__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator<= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const *__rhs)`

- `template<typename _Bi_iter >`
`bool std::operator<= (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator<= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Bilter >`
`bool std::operator== (const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)`
- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
`bool std::operator== (const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
`bool std::operator== (const sub_match< _Bi_iter > &__lhs, const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator== (typename iterator_traits< _Bi_iter >::value_type const *__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator== (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const *__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator== (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator== (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Bi_iter, typename _Alloc >`
`bool std::operator== (const match_results< _Bi_iter, _Alloc > &__m1, const match_results< _Bi_iter, _Alloc > &__m2)`
- `template<typename _Bilter >`
`bool std::operator> (const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)`
- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
`bool std::operator> (const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter, class _Ch_traits, class _Ch_alloc >`
`bool std::operator> (const sub_match< _Bi_iter > &__lhs, const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator> (typename iterator_traits< _Bi_iter >::value_type const *__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator> (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const *__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator> (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator> (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Bilter >`
`bool std::operator>= (const sub_match< _Bilter > &__lhs, const sub_match< _Bilter > &__rhs)`
- `template<typename _Bi_iter, typename _Ch_traits, typename _Ch_alloc >`
`bool std::operator>= (const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__lhs, const sub_match< _Bi_iter > &__rhs)`

- `template<typename _Bi_iter, class _Ch_traits, class _Ch_alloc >`
`bool std::operator>= (const sub_match< _Bi_iter > &__lhs, const __sub_match_string< _Bi_iter, _Ch_traits, _Ch_alloc > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator>= (typename iterator_traits< _Bi_iter >::value_type const *__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator>= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const *__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator>= (typename iterator_traits< _Bi_iter >::value_type const &__lhs, const sub_match< _Bi_iter > &__rhs)`
- `template<typename _Bi_iter >`
`bool std::operator>= (const sub_match< _Bi_iter > &__lhs, typename iterator_traits< _Bi_iter >::value_type const &__rhs)`
- `template<typename _Ch_type, typename _Rx_traits >`
`void std::swap (basic_regex< _Ch_type, _Rx_traits > &__lhs, basic_regex< _Ch_type, _Rx_traits > &__rhs)`
- `template<typename _Bi_iter, typename _Alloc >`
`void std::swap (match_results< _Bi_iter, _Alloc > &__lhs, match_results< _Bi_iter, _Alloc > &__rhs)`

Matching, Searching, and Replacing

- `template<typename _Bi_iter, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_match (_Bi_iter __s, _Bi_iter __e, match_results< _Bi_iter, _Alloc > &__m, const basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Bi_iter, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_match (_Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Ch_type, typename _Alloc, typename _Rx_traits >`
`bool std::regex_match (const _Ch_type *__s, match_results< const _Ch_type *, _Alloc > &__m, const basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_match (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &__s, match_results< typename basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &__m, const basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_match (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &&, match_results< typename basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &, const basic_regex< _Ch_type, _Rx_traits > &, regex_constants::match_flag_type=regex_constants::match_default)=delete`
- `template<typename _Ch_type, class _Rx_traits >`
`bool std::regex_match (const _Ch_type *__s, const basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Str_allocator, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_match (const basic_string< _Ch_type, _Ch_traits, _Str_allocator > &__s, const basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Bi_iter, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (_Bi_iter __s, _Bi_iter __e, match_results< _Bi_iter, _Alloc > &__m, const basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Bi_iter, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (_Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type, _Rx_traits > &__re, regex_constants::match_flag_type __flags=regex_constants::match_default)`

- `template<typename _Ch_type, class _Alloc, class _Rx_traits >`
`bool std::regex_search (const _Ch_type *__s, match_results< const _Ch_type *, _Alloc > &__m, const`
`basic_regex< _Ch_type, _Rx_traits > &__e, regex_constants::match_flag_type __f=regex_constants::`
`::match_default)`
- `template<typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (const _Ch_type *__s, const basic_regex< _Ch_type, _Rx_traits > &__e, regex_`
`constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _String_allocator, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (const basic_string< _Ch_type, _Ch_traits, _String_allocator > &__s, const basic_`
`regex< _Ch_type, _Rx_traits > &__e, regex_constants::match_flag_type __flags=regex_constants::match_`
`default)`
- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &__s, match_results< type-`
`name basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &__m, const basic_regex<`
`_Ch_type, _Rx_traits > &__e, regex_constants::match_flag_type __f=regex_constants::match_default)`
- `template<typename _Ch_traits, typename _Ch_alloc, typename _Alloc, typename _Ch_type, typename _Rx_traits >`
`bool std::regex_search (const basic_string< _Ch_type, _Ch_traits, _Ch_alloc > &__, match_results< type-`
`name basic_string< _Ch_type, _Ch_traits, _Ch_alloc >::const_iterator, _Alloc > &&, const basic_regex< _`
`Ch_type, _Rx_traits > &, regex_constants::match_flag_type=regex_constants::match_default)=delete`
- `template<typename _Out_iter, typename _Bi_iter, typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa >`
`_Out_iter std::regex_replace (_Out_iter __out, _Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type,`
`_Rx_traits > &__e, const basic_string< _Ch_type, _St, _Sa > &__fmt, regex_constants::match_flag_type`
`__flags=regex_constants::match_default)`
- `template<typename _Out_iter, typename _Bi_iter, typename _Rx_traits, typename _Ch_type >`
`_Out_iter std::regex_replace (_Out_iter __out, _Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type,`
`_Rx_traits > &__e, const _Ch_type *__fmt, regex_constants::match_flag_type __flags=regex_constants::`
`::match_default)`
- `template<typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa, typename _Fst, typename _Fsa >`
`basic_string< _Ch_type, _St, _Sa > std::regex_replace (const basic_string< _Ch_type, _St, _Sa > &__s,`
`const basic_regex< _Ch_type, _Rx_traits > &__e, const basic_string< _Ch_type, _Fst, _Fsa > &__fmt,`
`regex_constants::match_flag_type __flags=regex_constants::match_default)`
- `template<typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa >`
`basic_string< _Ch_type, _St, _Sa > std::regex_replace (const basic_string< _Ch_type, _St, _Sa > &__s,`
`const basic_regex< _Ch_type, _Rx_traits > &__e, const _Ch_type *__fmt, regex_constants::match_flag_`
`type __flags=regex_constants::match_default)`
- `template<typename _Rx_traits, typename _Ch_type, typename _St, typename _Sa >`
`basic_string< _Ch_type > std::regex_replace (const _Ch_type *__s, const basic_regex< _Ch_type, _Rx_`
`_traits > &__e, const basic_string< _Ch_type, _St, _Sa > &__fmt, regex_constants::match_flag_type __`
`flags=regex_constants::match_default)`
- `template<typename _Rx_traits, typename _Ch_type >`
`basic_string< _Ch_type > std::regex_replace (const _Ch_type *__s, const basic_regex< _Ch_type, _Rx_`
`_traits > &__e, const _Ch_type *__fmt, regex_constants::match_flag_type __flags=regex_constants::match_`
`default)`

6.480.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

6.481 regex.tcc File Reference

Namespaces

- [std](#)
- [std::__detail](#)

Functions

- `template<typename _Bilter, typename _Alloc, typename _CharT, typename _TraitsT, _RegexExecutorPolicy __policy, bool __match_↵
mode>
bool std::__detail::__regex_algo_impl (_Bilter __s, _Bilter __e, match_results< _Bilter, _Alloc > &__m, const
basic_regex< _CharT, _TraitsT > &__re, regex_constants::match_flag_type __flags)`

Matching, Searching, and Replacing

- `template<typename _Out_iter, typename _Bi_iter, typename _Rx_traits, typename _Ch_type >
_Out_iter std::regex_replace (_Out_iter __out, _Bi_iter __first, _Bi_iter __last, const basic_regex< _Ch_type,
_Rx_traits > &__e, const _Ch_type *__fmt, regex_constants::match_flag_type __flags=regex_constants::↵
::match_default)`

6.481.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

6.482 `regex_automaton.h` File Reference

Classes

- class `std::__detail::StateSeq< _TraitsT >`

Namespaces

- `std`
- `std::__detail`

Macros

- `#define _GLIBCXX_REGEX_STATE_LIMIT`

Typedefs

- `template<typename _CharT >
using std::__detail::Matcher = std::function< bool(_CharT)>`
- `typedef long std::__detail::StateldT`

Enumerations

- `enum std::__detail::Opcode : int {
_S_opcode_unknown, _S_opcode_alternative, _S_opcode_repeat, _S_opcode_backref,
_S_opcode_line_begin_assertion, _S_opcode_line_end_assertion, _S_opcode_word_boundary, _S_↵
opcode_subexpr_lookahead,
_S_opcode_subexpr_begin, _S_opcode_subexpr_end, _S_opcode_dummy, _S_opcode_match,
_S_opcode_accept }`

Variables

- static const `_StateIdT` `std::__detail::_S_invalid_state_id`

6.482.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

6.483 `regex_automaton.tcc` File Reference

Namespaces

- [std](#)
- [std::__detail](#)

6.483.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

6.484 `regex_compiler.h` File Reference

Classes

- struct [std::__detail::BracketMatcher](#)< `_TraitsT`, `__icase`, `__collate` >
- struct [std::__detail::BracketMatcher](#)< `_TraitsT`, `__icase`, `__collate` >
- class [std::__detail::Compiler](#)< `_TraitsT` >

Namespaces

- [std](#)
- [std::__detail](#)

Typedefs

- template<typename `_Iter` , typename `_TraitsT` >
using [std::__detail::__disable_if_contiguous_normal_iter](#) = typename enable_if< `!__is_contiguous_normal_iter`< `_Iter` >::value, [std::shared_ptr](#)< const `_NFA`< `_TraitsT` >> >::type
- template<typename `_Iter` , typename `_TraitsT` >
using [std::__detail::__enable_if_contiguous_normal_iter](#) = typename enable_if< `__is_contiguous_normal_iter`< `_Iter` >::value, [std::shared_ptr](#)< const `_NFA`< `_TraitsT` >> >::type

Functions

- `template<typename _FwdIter, typename _TraitsT > std::__detail::__compile_nfa (_FwdIter __first, ↵
_FwdIter __last, const typename _TraitsT::locale_type &__loc, regex_constants::syntax_option_type __flags)`
- `template<typename _FwdIter, typename _TraitsT > std::__detail::__compile_nfa (_FwdIter __first, ↵
_FwdIter __last, const typename _TraitsT::locale_type &__loc, regex_constants::syntax_option_type __flags)`

6.484.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

6.485 regex_compiler.tcc File Reference

Namespaces

- [std](#)
- [std::__detail](#)

Macros

- `#define __INSERT_REGEX_MATCHER(__func, args...)`

6.485.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

6.486 regex_constants.h File Reference

Namespaces

- [std](#)
- [std::regex_constants](#)

5.1 Regular Expression Syntax Options

- enum `std::regex_constants::__syntax_option` {
`_S_icode`, `_S_nosubs`, `_S_optimize`, `_S_collate`,
`_S_ECMAScript`, `_S_basic`, `_S_extended`, `_S_awk`,
`_S_grep`, `_S_egrep`, `_S_polynomial`, `_S_syntax_last` }
- enum `std::regex_constants::syntax_option_type` : unsigned int
- constexpr `syntax_option_type` `std::regex_constants::icase`
- constexpr `syntax_option_type` `std::regex_constants::nosubs`
- constexpr `syntax_option_type` `std::regex_constants::optimize`
- constexpr `syntax_option_type` `std::regex_constants::collate`
- constexpr `syntax_option_type` `std::regex_constants::ECMAScript`
- constexpr `syntax_option_type` `std::regex_constants::basic`
- constexpr `syntax_option_type` `std::regex_constants::extended`
- constexpr `syntax_option_type` `std::regex_constants::awk`
- constexpr `syntax_option_type` `std::regex_constants::grep`
- constexpr `syntax_option_type` `std::regex_constants::egrep`
- constexpr `syntax_option_type` `std::regex_constants::__polynomial`
- constexpr `syntax_option_type` `std::regex_constants::operator&` (`syntax_option_type __a`, `syntax_option_type __b`)
- constexpr `syntax_option_type` `std::regex_constants::operator|` (`syntax_option_type __a`, `syntax_option_type __b`)
- constexpr `syntax_option_type` `std::regex_constants::operator^` (`syntax_option_type __a`, `syntax_option_type __b`)
- constexpr `syntax_option_type` `std::regex_constants::operator~` (`syntax_option_type __a`)
- `syntax_option_type &` `std::regex_constants::operator&=` (`syntax_option_type & __a`, `syntax_option_type __b`)
- `syntax_option_type &` `std::regex_constants::operator|=` (`syntax_option_type & __a`, `syntax_option_type __b`)
- `syntax_option_type &` `std::regex_constants::operator^=` (`syntax_option_type & __a`, `syntax_option_type __b`)

5.2 Matching Rules

Matching a regular expression against a sequence of characters [first, last) proceeds according to the rules of the grammar specified for the regular expression object, modified according to the effects listed below for any bitmask elements set.

- enum `std::regex_constants::__match_flag` {
`_S_not_bol`, `_S_not_eol`, `_S_not_bow`, `_S_not_eow`,
`_S_any`, `_S_not_null`, `_S_continuous`, `_S_prev_avail`,
`_S_sed`, `_S_no_copy`, `_S_first_only`, `_S_match_flag_last` }
- enum `std::regex_constants::match_flag_type` : unsigned int
- constexpr `match_flag_type` `std::regex_constants::match_default`
- constexpr `match_flag_type` `std::regex_constants::match_not_bol`
- constexpr `match_flag_type` `std::regex_constants::match_not_eol`
- constexpr `match_flag_type` `std::regex_constants::match_not_bow`
- constexpr `match_flag_type` `std::regex_constants::match_not_eow`
- constexpr `match_flag_type` `std::regex_constants::match_any`
- constexpr `match_flag_type` `std::regex_constants::match_not_null`
- constexpr `match_flag_type` `std::regex_constants::match_continuous`
- constexpr `match_flag_type` `std::regex_constants::match_prev_avail`
- constexpr `match_flag_type` `std::regex_constants::format_default`
- constexpr `match_flag_type` `std::regex_constants::format_sed`

- constexpr match_flag_type [std::regex_constants::format_no_copy](#)
- constexpr match_flag_type [std::regex_constants::format_first_only](#)
- constexpr match_flag_type [std::regex_constants::operator&](#) (match_flag_type __a, match_flag_type __b)
- constexpr match_flag_type [std::regex_constants::operator|](#) (match_flag_type __a, match_flag_type __b)
- constexpr match_flag_type [std::regex_constants::operator^](#) (match_flag_type __a, match_flag_type __b)
- constexpr match_flag_type [std::regex_constants::operator~](#) (match_flag_type __a)
- match_flag_type & [std::regex_constants::operator&=](#) (match_flag_type & __a, match_flag_type __b)
- match_flag_type & [std::regex_constants::operator|=](#) (match_flag_type & __a, match_flag_type __b)
- match_flag_type & [std::regex_constants::operator^=](#) (match_flag_type & __a, match_flag_type __b)

6.486.1 Detailed Description

Constant definitions for the std regex library.

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

6.487 `regex_error.h` File Reference

Classes

- class [std::regex_error](#)

Namespaces

- [std](#)
- [std::regex_constants](#)

Functions

- void **std::**[__throw_regex_error](#) (regex_constants::error_type __ecode)
- void **std::**[__throw_regex_error](#) (regex_constants::error_type __ecode, const char * __what)

5.3 Error Types

- enum [std::regex_constants::error_type](#) {
[_S_error_collate](#), [_S_error_ctype](#), [_S_error_escape](#), [_S_error_backref](#),
[_S_error_brack](#), [_S_error_paren](#), [_S_error_brace](#), [_S_error_badbrace](#),
[_S_error_range](#), [_S_error_space](#), [_S_error_badrepeat](#), [_S_error_complexity](#),
[_S_error_stack](#) }
- constexpr error_type [std::regex_constants::error_collate](#) ([_S_error_collate](#))
- constexpr error_type [std::regex_constants::error_ctype](#) ([_S_error_ctype](#))
- constexpr error_type [std::regex_constants::error_escape](#) ([_S_error_escape](#))
- constexpr error_type [std::regex_constants::error_backref](#) ([_S_error_backref](#))
- constexpr error_type [std::regex_constants::error_brack](#) ([_S_error_brack](#))
- constexpr error_type [std::regex_constants::error_paren](#) ([_S_error_paren](#))
- constexpr error_type [std::regex_constants::error_brace](#) ([_S_error_brace](#))
- constexpr error_type [std::regex_constants::error_badbrace](#) ([_S_error_badbrace](#))
- constexpr error_type [std::regex_constants::error_range](#) ([_S_error_range](#))
- constexpr error_type [std::regex_constants::error_space](#) ([_S_error_space](#))
- constexpr error_type [std::regex_constants::error_badrepeat](#) ([_S_error_badrepeat](#))
- constexpr error_type [std::regex_constants::error_complexity](#) ([_S_error_complexity](#))
- constexpr error_type [std::regex_constants::error_stack](#) ([_S_error_stack](#))

6.487.1 Detailed Description

Error and exception objects for the `std::regex` library.

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

6.488 `regex_executor.h` File Reference

Classes

- class [std::__detail::_Executor<_Bilter, _Alloc, _TraitsT, __dfs_mode>](#)

Namespaces

- [std](#)
- [std::__detail](#)

6.488.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

6.489 `regex_executor.tcc` File Reference

Namespaces

- [std](#)
- [std::__detail](#)

6.489.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

6.490 `regex_scanner.h` File Reference

Classes

- class [std::__detail::_Scanner<_CharT>](#)

Namespaces

- [std](#)
- [std::__detail](#)

6.490.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

6.491 `regex_scanner.tcc` File Reference

Namespaces

- [std](#)
- [std::__detail](#)

6.491.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<regex>`.

6.492 `resize_fn_imps.hpp` File Reference

6.492.1 Detailed Description

Contains implementations of `cc_ht_map_`'s resize related functions.

6.493 `resize_fn_imps.hpp` File Reference

6.493.1 Detailed Description

Contains implementations of `gp_ht_map_`'s resize related functions.

6.494 `resize_no_store_hash_fn_imps.hpp` File Reference

6.494.1 Detailed Description

Contains implementations of `cc_ht_map_`'s resize related functions, when the hash value is not stored.

6.495 `resize_no_store_hash_fn_imps.hpp` File Reference

6.495.1 Detailed Description

Contains implementations of `gp_ht_map_`'s resize related functions, when the hash value is not stored.

6.496 `resize_policy.hpp` File Reference

Classes

- class [__gnu_pbds::detail::resize_policy< _Tp >](#)

Namespaces

- [__gnu_pbds](#)

6.496.1 Detailed Description

Contains an implementation class for a `binary_heap`.

6.497 `resize_store_hash_fn_imps.hpp` File Reference

6.497.1 Detailed Description

Contains implementations of `cc_ht_map_`'s resize related functions, when the hash value is stored.

6.498 `resize_store_hash_fn_imps.hpp` File Reference

6.498.1 Detailed Description

Contains implementations of `gp_ht_map_`'s resize related functions, when the hash value is stored.

6.499 `rope` File Reference

Classes

- class [__gnu_cxx::rope< _CharT, _Alloc >](#)
- class [__gnu_cxx::rope< _CharT, _Alloc >](#)

Namespaces

- [__gnu_cxx](#)
- [__gnu_cxx::__detail](#)
- [std](#)
- [std::tr1](#)

Macros

- `#define __GC_CONST`
- `#define __ROPE_DEFINE_ALLOC(_Tp, __name)`
- `#define __ROPE_DEFINE_ALLOC(_Tp, __name)`
- `#define __ROPE_DEFINE_ALLOCS(__a)`
- `#define __STATIC_IF_SGI_ALLOC`
- `#define __STL_FREE_STRING(__s, __l, __a)`
- `#define __STL_ROPE_FROM_UNOWNED_CHAR_PTR(__s, __size, __a)`
- `#define _ROPE`

Typedefs

- `typedef rope< char > __gnu_cxx::crope`
- `typedef rope< wchar_t > __gnu_cxx::wrope`

Enumerations

- `enum { _S_max_rope_depth }`
- `enum _Tag { _S_leaf, _S_concat, _S_substringfn, _S_function }`

Functions

- `crope::reference __gnu_cxx::__mutable_reference_at (crope &__c, size_t __i)`
- `template<typename _ForwardIterator, typename _Allocator >`
`void __gnu_cxx::Destroy_const (_ForwardIterator __first, _ForwardIterator __last, _Allocator __alloc)`
- `template<typename _ForwardIterator, typename _Tp >`
`void __gnu_cxx::Destroy_const (_ForwardIterator __first, _ForwardIterator __last, allocator< _Tp >)`
- `template<class _CharT >`
`void __gnu_cxx::S_cond_store_eos (_CharT &)`
- `void __gnu_cxx::S_cond_store_eos (char &__c)`
- `void __gnu_cxx::S_cond_store_eos (wchar_t &__c)`
- `template<class _CharT >`
`_CharT __gnu_cxx::S_eos (_CharT *)`
- `template<class _CharT >`
`bool __gnu_cxx::S_is_basic_char_type (_CharT *)`
- `bool __gnu_cxx::S_is_basic_char_type (char *)`
- `bool __gnu_cxx::S_is_basic_char_type (wchar_t *)`
- `template<class _CharT >`
`bool __gnu_cxx::S_is_one_byte_char_type (_CharT *)`
- `bool __gnu_cxx::S_is_one_byte_char_type (char *)`

- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator!= (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator!= (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator!= (const rope< _CharT, _Alloc > &__x, const rope< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator!= (const _Rope_char_ptr_proxy< _CharT, _Alloc > &__x, const _Rope_char_ptr_proxy< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`_Rope_const_iterator< _CharT, _Alloc > __gnu_cxx::operator+ (const _Rope_const_iterator< _CharT, _Alloc > &__x, ptrdiff_t __n)`
- `template<class _CharT, class _Alloc >`
`_Rope_const_iterator< _CharT, _Alloc > __gnu_cxx::operator+ (ptrdiff_t __n, const _Rope_const_iterator< _CharT, _Alloc > &__x)`
- `template<class _CharT, class _Alloc >`
`_Rope_iterator< _CharT, _Alloc > __gnu_cxx::operator+ (const _Rope_iterator< _CharT, _Alloc > &__x, ptrdiff_t __n)`
- `template<class _CharT, class _Alloc >`
`_Rope_iterator< _CharT, _Alloc > __gnu_cxx::operator+ (ptrdiff_t __n, const _Rope_iterator< _CharT, _Alloc > &__x)`
- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > __gnu_cxx::operator+ (const rope< _CharT, _Alloc > &__left, const rope< _CharT, _Alloc > &__right)`
- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > __gnu_cxx::operator+ (const rope< _CharT, _Alloc > &__left, const _CharT *__right)`
- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > __gnu_cxx::operator+ (const rope< _CharT, _Alloc > &__left, _CharT __right)`
- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > & __gnu_cxx::operator+= (rope< _CharT, _Alloc > &__left, const rope< _CharT, _Alloc > &__right)`
- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > & __gnu_cxx::operator+= (rope< _CharT, _Alloc > &__left, const _CharT *__right)`
- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > & __gnu_cxx::operator+= (rope< _CharT, _Alloc > &__left, _CharT __right)`
- `template<class _CharT, class _Alloc >`
`_Rope_const_iterator< _CharT, _Alloc > __gnu_cxx::operator- (const _Rope_const_iterator< _CharT, _Alloc > &__x, ptrdiff_t __n)`
- `template<class _CharT, class _Alloc >`
`ptrdiff_t __gnu_cxx::operator- (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`_Rope_iterator< _CharT, _Alloc > __gnu_cxx::operator- (const _Rope_iterator< _CharT, _Alloc > &__x, ptrdiff_t __n)`
- `template<class _CharT, class _Alloc >`
`ptrdiff_t __gnu_cxx::operator- (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator< (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`

- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator< (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator< (const rope< _CharT, _Alloc > &__left, const rope< _CharT, _Alloc > &__right)`
- `template<class _CharT, class _Traits, class _Alloc >`
`std::basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (std::basic_ostream< _CharT, _Traits > &__o, const rope< _CharT, _Alloc > &__r)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator<= (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator<= (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator<= (const rope< _CharT, _Alloc > &__x, const rope< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator== (const _Rope_char_ptr_proxy< _CharT, _Alloc > &__x, const _Rope_char_ptr_proxy< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator== (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator== (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator== (const rope< _CharT, _Alloc > &__left, const rope< _CharT, _Alloc > &__right)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator> (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator> (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator> (const rope< _CharT, _Alloc > &__x, const rope< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator>= (const _Rope_const_iterator< _CharT, _Alloc > &__x, const _Rope_const_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator>= (const _Rope_iterator< _CharT, _Alloc > &__x, const _Rope_iterator< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`bool __gnu_cxx::operator>= (const rope< _CharT, _Alloc > &__x, const rope< _CharT, _Alloc > &__y)`
- `template<class _CharT, class _Alloc >`
`void __gnu_cxx::swap (_Rope_char_ref_proxy< _CharT, _Alloc > __a, _Rope_char_ref_proxy< _CharT, _Alloc > __b)`
- `template<class _CharT, class _Alloc >`
`void __gnu_cxx::swap (rope< _CharT, _Alloc > &__x, rope< _CharT, _Alloc > &__y)`

Variables

- `template<class _CharT, class _Alloc >`
`rope< _CharT, _Alloc > __gnu_cxx::identity_element (_Rope_Concat_fn< _CharT, _Alloc >)`

6.499.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGI STL subset).

6.500 ropeimpl.h File Reference

Namespaces

- [__gnu_cxx](#)

Functions

- `template<class _CharT, class _Traits >`
`void __gnu_cxx::Rope_fill (basic_ostream< _CharT, _Traits > &__o, size_t __n)`
- `template<class _CharT >`
`bool __gnu_cxx::Rope_is_simple (_CharT *)`
- `bool __gnu_cxx::Rope_is_simple (char *)`
- `bool __gnu_cxx::Rope_is_simple (wchar_t *)`
- `template<class _Rope_iterator >`
`void __gnu_cxx::Rope_rotate (_Rope_iterator __first, _Rope_iterator __middle, _Rope_iterator __last)`
- `template<class _CharT, class _Traits, class _Alloc >`
`basic_ostream< _CharT, _Traits > & __gnu_cxx::operator<< (basic_ostream< _CharT, _Traits > &__o, const rope< _CharT, _Alloc > &__r)`
- `void __gnu_cxx::rotate (_Rope_iterator< char, __STL_DEFAULT_ALLOCATOR(char)> __first, _Rope_iterator< char, __STL_DEFAULT_ALLOCATOR(char)> __middle, _Rope_iterator< char, __STL_DEFAULT_ALLOCATOR(char)> __last)`

6.500.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ext/rope>`.

6.501 rotate_fn_imps.hpp File Reference

6.501.1 Detailed Description

Contains imps for rotating nodes.

6.502 rotate_fn_imps.hpp File Reference

6.502.1 Detailed Description

Contains imps for rotating nodes.

6.503 `safe_base.h` File Reference

Classes

- class [__gnu_debug::_Safe_iterator_base](#)
- class [__gnu_debug::_Safe_sequence_base](#)

Namespaces

- [__gnu_debug](#)

Functions

- bool [__gnu_debug::__check_singular_aux](#) (const [_Safe_iterator_base](#) *__x)

6.503.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.504 `safe_container.h` File Reference

Classes

- class [__gnu_debug::_Safe_container<_SafeContainer, _Alloc, _SafeBase, _IsCxx11AllocatorAware >](#)

Namespaces

- [__gnu_debug](#)

6.504.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.505 `safe_iterator.h` File Reference

Classes

- struct [__gnu_debug::_BeforeBeginHelper<_Sequence >](#)
- class [__gnu_debug::_Safe_iterator<_Iterator, _Sequence >](#)
- struct [__gnu_debug::_Sequence_traits<_Sequence >](#)

Namespaces

- [__gnu_debug](#)

Functions

- `template<typename _Iterator, typename _Sequence >
_Iterator __gnu_debug::__base (const _Safe_iterator< _Iterator, _Sequence > &__it, std::random_access_iterator_tag)`
- `template<typename _Iterator, typename _Sequence >
const _Safe_iterator< _Iterator, _Sequence > & __gnu_debug::__base (const _Safe_iterator< _Iterator, std::input_iterator_tag`
- `template<typename _Iterator, typename _Sequence >
auto __gnu_debug::__base (const _Safe_iterator< _Iterator, _Sequence > &__it) -> decltype(__base(__it, std::__iterator_category(__it)))`
- `template<typename _Iterator, typename _Sequence >
bool __gnu_debug::__check_dereferenceable (const _Safe_iterator< _Iterator, _Sequence > &__x)`
- `template<typename _Iterator, typename _Sequence >
_Distance_traits< _Iterator >::__type __gnu_debug::__get_distance (const _Safe_iterator< _Iterator, std::random_access_iterator_tag`
- `template<typename _Iterator, typename _Sequence >
_Distance_traits< _Iterator >::__type __gnu_debug::__get_distance (const _Safe_iterator< _Iterator, std::input_iterator_tag`
- `template<typename _Iterator, typename _Sequence >
_Distance_traits< _Iterator >::__type __gnu_debug::__get_distance_from_begin (const _Safe_iterator< __it`
- `template<typename _Iterator, typename _Sequence >
_Distance_traits< _Iterator >::__type __gnu_debug::__get_distance_to_end (const _Safe_iterator< _Iterator, __it)`
- `template<typename _Iterator, typename _Sequence >
_Iterator __gnu_debug::__unsafe (const _Safe_iterator< _Iterator, _Sequence > &__it)`
- `template<typename _Iterator, typename _Sequence >
bool __gnu_debug::__valid_range (const _Safe_iterator< _Iterator, _Sequence > &__first, const _Safe_iterator< __last, typename _Distance_traits< _Iterator >::__type &__dist)`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >
bool __gnu_debug::operator!= (const _Safe_iterator< _IteratorL, _Sequence > &__lhs, const _Safe_iterator< __rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >
bool __gnu_debug::operator!= (const _Safe_iterator< _Iterator, _Sequence > &__lhs, const _Safe_iterator< __rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >
_Safe_iterator< _Iterator, _Sequence > __gnu_debug::operator+ (typename _Safe_iterator< _Iterator, __n, const _Safe_iterator< _Iterator, _Sequence > &__i) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >
_Safe_iterator< _IteratorL, _Sequence >::__difference_type __gnu_debug::operator- (const _Safe_iterator< __lhs, const _Safe_iterator< _IteratorR, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >
_Safe_iterator< _Iterator, _Sequence >::__difference_type __gnu_debug::operator- (const _Safe_iterator< __lhs, const _Safe_iterator< _Iterator, _Sequence > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >
bool __gnu_debug::operator< (const _Safe_iterator< _IteratorL, _Sequence > &__lhs, const _Safe_iterator< __rhs) noexcept`

- `template<typename _Iterator, typename _Sequence >`
`bool __gnu_debug::operator< (const _Safe_iterator< _Iterator, _Sequence > &__lhs, const _Safe_iterator<`
`_Iterator, _Sequence > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`bool __gnu_debug::operator<= (const _Safe_iterator< _IteratorL, _Sequence > &__lhs, const _Safe_iterator<`
`_IteratorR, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`bool __gnu_debug::operator<= (const _Safe_iterator< _Iterator, _Sequence > &__lhs, const _Safe_iterator<`
`_Iterator, _Sequence > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`bool __gnu_debug::operator== (const _Safe_iterator< _IteratorL, _Sequence > &__lhs, const _Safe_iterator<`
`_IteratorR, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`bool __gnu_debug::operator== (const _Safe_iterator< _Iterator, _Sequence > &__lhs, const _Safe_iterator<`
`_Iterator, _Sequence > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`bool __gnu_debug::operator> (const _Safe_iterator< _IteratorL, _Sequence > &__lhs, const _Safe_iterator<`
`_IteratorR, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`bool __gnu_debug::operator> (const _Safe_iterator< _Iterator, _Sequence > &__lhs, const _Safe_iterator<`
`_Iterator, _Sequence > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR, typename _Sequence >`
`bool __gnu_debug::operator>= (const _Safe_iterator< _IteratorL, _Sequence > &__lhs, const _Safe_iterator<`
`_IteratorR, _Sequence > &__rhs) noexcept`
- `template<typename _Iterator, typename _Sequence >`
`bool __gnu_debug::operator>= (const _Safe_iterator< _Iterator, _Sequence > &__lhs, const _Safe_iterator<`
`_Iterator, _Sequence > &__rhs) noexcept`

6.505.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.506 safe_iterator.tcc File Reference

Namespaces

- [__gnu_debug](#)

Macros

- `#define _GLIBCXX_DEBUG_SAFE_ITERATOR_TCC`

6.506.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.507 `safe_local_iterator.h` File Reference

Classes

- class [__gnu_debug::Safe_local_iterator](#)< `_Iterator`, `_Sequence` >

Namespaces

- [__gnu_debug](#)

Functions

- template<typename `_Iterator` , typename `_Sequence` >
bool [__gnu_debug::check_dereferenceable](#) (const `_Safe_local_iterator`< `_Iterator`, `_Sequence` > &__x)
- template<typename `_Iterator` , typename `_Sequence` >
[std::pair](#)< typename `std::iterator_traits`< `_Iterator` >::difference_type, `_Distance_precision` > [__gnu_debug::get_distance](#) (const `_Safe_local_iterator`< `_Iterator`, `_Sequence` > &__first, const `_Safe_local_iterator`< `_Iterator`, `_Sequence` > &__last, [std::input_iterator_tag](#))
- template<typename `_Iterator` , typename `_Sequence` >
`_Iterator` [__gnu_debug::unsafe](#) (const `_Safe_local_iterator`< `_Iterator`, `_Sequence` > &__it)
- template<typename `_Iterator` , typename `_Sequence` >
bool [__gnu_debug::valid_range](#) (const `_Safe_local_iterator`< `_Iterator`, `_Sequence` > &__first, const `_Safe_local_iterator`< `_Iterator`, `_Sequence` > &__last, typename `_Distance_traits`< `_Iterator` >::__type &__dist_info)
- template<typename `_IteratorL` , typename `_IteratorR` , typename `_Sequence` >
bool [__gnu_debug::operator!=](#) (const `_Safe_local_iterator`< `_IteratorL`, `_Sequence` > &__lhs, const `_Safe_local_iterator`< `_IteratorR`, `_Sequence` > &__rhs)
- template<typename `_Iterator` , typename `_Sequence` >
bool [__gnu_debug::operator!=](#) (const `_Safe_local_iterator`< `_Iterator`, `_Sequence` > &__lhs, const `_Safe_local_iterator`< `_Iterator`, `_Sequence` > &__rhs)
- template<typename `_IteratorL` , typename `_IteratorR` , typename `_Sequence` >
bool [__gnu_debug::operator==](#) (const `_Safe_local_iterator`< `_IteratorL`, `_Sequence` > &__lhs, const `_Safe_local_iterator`< `_IteratorR`, `_Sequence` > &__rhs)
- template<typename `_Iterator` , typename `_Sequence` >
bool [__gnu_debug::operator==](#) (const `_Safe_local_iterator`< `_Iterator`, `_Sequence` > &__lhs, const `_Safe_local_iterator`< `_Iterator`, `_Sequence` > &__rhs)

6.507.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.508 `safe_local_iterator.tcc` File Reference

Namespaces

- [__gnu_debug](#)

Macros

- `#define _GLIBCXX_DEBUG_SAFE_LOCAL_ITERATOR_TCC`

6.508.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.509 `safe_sequence.h` File Reference

Classes

- class [__gnu_debug::_After_nth_from<_Iterator>](#)
- class [__gnu_debug::_Equal_to<_Type>](#)
- class [__gnu_debug::_Not_equal_to<_Type>](#)
- class [__gnu_debug::_Safe_node_sequence<_Sequence>](#)
- class [__gnu_debug::_Safe_sequence<_Sequence>](#)

Namespaces

- [__gnu_debug](#)

6.509.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.510 `safe_sequence.tcc` File Reference

Namespaces

- [__gnu_debug](#)

Macros

- `#define _GLIBCXX_DEBUG_SAFE_SEQUENCE_TCC`

6.510.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.511 `safe_unordered_base.h` File Reference

Classes

- class [__gnu_debug::_Safe_local_iterator_base](#)
- class [__gnu_debug::_Safe_unordered_container_base](#)

Namespaces

- [__gnu_debug](#)

6.511.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.512 `safe_unordered_container.h` File Reference

Classes

- class [__gnu_debug::_Safe_unordered_container<_Container>](#)

Namespaces

- [__gnu_debug](#)

6.512.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.513 `safe_unordered_container.tcc` File Reference

Namespaces

- [__gnu_debug](#)

Macros

- `#define _GLIBCXX_DEBUG_SAFE_UNORDERED_CONTAINER_TCC`

6.513.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.514 `sample_probe_fn.hpp` File Reference

Classes

- class [__gnu_pbds::sample_probe_fn](#)

Namespaces

- [__gnu_pbds](#)

6.514.1 Detailed Description

Contains a sample probe policy.

6.515 `sample_range_hashing.hpp` File Reference

Classes

- class [__gnu_pbds::sample_range_hashing](#)

Namespaces

- [__gnu_pbds](#)

6.515.1 Detailed Description

Contains a range hashing policy.

6.516 `sample_ranged_hash_fn.hpp` File Reference

Classes

- class [__gnu_pbds::sample_ranged_hash_fn](#)

Namespaces

- [__gnu_pbds](#)

6.516.1 Detailed Description

Contains a ranged hash policy.

6.517 `sample_ranged_probe_fn.hpp` File Reference

Classes

- class [__gnu_pbds::sample_ranged_probe_fn](#)

Namespaces

- [__gnu_pbds](#)

6.517.1 Detailed Description

Contains a ranged probe policy.

6.518 `sample_resize_policy.hpp` File Reference

Classes

- class [__gnu_pbds::sample_resize_policy](#)

Namespaces

- [__gnu_pbds](#)

6.518.1 Detailed Description

Contains a sample resize policy for hash tables.

6.519 `sample_resize_trigger.hpp` File Reference

Classes

- class [__gnu_pbds::sample_resize_trigger](#)

Namespaces

- [__gnu_pbds](#)

6.519.1 Detailed Description

Contains a sample resize trigger policy class.

6.520 `sample_size_policy.hpp` File Reference

Classes

- class [__gnu_pbds::sample_size_policy](#)

Namespaces

- [__gnu_pbds](#)

6.520.1 Detailed Description

Contains a sample size resize-policy.

6.521 `sample_tree_node_update.hpp` File Reference

Classes

- class [__gnu_pbds::sample_tree_node_update< Const_Node_Iter, Node_Iter, Cmp_Fn, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

6.521.1 Detailed Description

Contains a samle node update functor.

6.522 `sample_trie_access_traits.hpp` File Reference

Classes

- struct [__gnu_pbds::sample_trie_access_traits](#)

Namespaces

- [__gnu_pbds](#)

6.522.1 Detailed Description

Contains a sample probe policy.

6.523 sample_trie_node_update.hpp File Reference

Classes

- class [__gnu_pbds::sample_trie_node_update](#)< Node_Cltr, Node_Itr, _ATraits, _Alloc >

Namespaces

- [__gnu_pbds](#)

6.523.1 Detailed Description

Contains a samle node update functor.

6.524 sample_update_policy.hpp File Reference

Classes

- struct [__gnu_pbds::sample_update_policy](#)

Namespaces

- [__gnu_pbds](#)

6.524.1 Detailed Description

Contains a sample policy for list update containers.

6.525 scoped_allocator File Reference

Classes

- class [std::scoped_allocator_adaptor](#)< _OuterAlloc, _InnerAllocs >
- class [std::scoped_allocator_adaptor](#)< _OuterAlloc, _InnerAllocs >

Namespaces

- [std](#)

Macros

- `#define _SCOPED_ALLOCATOR`

Typedefs

- `template<typename _Alloc >`
using **std::__outer_allocator_t** = decltype(std::declval<_Alloc >().outer_allocator())

Functions

- `template<typename _Alloc >`
`__outermost_type<_Alloc >::type & std::__outermost (_Alloc &__a)`
- `template<typename _OutA1, typename _OutA2, typename... _InA>`
bool **std::operator!=** (const scoped_allocator_adaptor<_OutA1, _InA... > &__a, const scoped_allocator_adaptor<_OutA2, _InA... > &__b) noexcept
- `template<typename _OutA1, typename _OutA2, typename... _InA>`
bool **std::operator==** (const scoped_allocator_adaptor<_OutA1, _InA... > &__a, const scoped_allocator_adaptor<_OutA2, _InA... > &__b) noexcept

6.525.1 Detailed Description

This is a Standard C++ Library header.

6.526 search.h File Reference

Namespaces

- [__gnu_parallel](#)

Functions

- `template<typename _RAIter, typename _DifferenceTp >`
void [__gnu_parallel::__calc_borders](#) (_RAIter __elements, _DifferenceTp __length, _DifferenceTp *__off)
- `template<typename __RAIter1, typename __RAIter2, typename _Pred >`
[__RAIter1 __gnu_parallel::__search_template](#) (__RAIter1 __begin1, __RAIter1 __end1, __RAIter2 __begin2, __RAIter2 __end2, _Pred __pred)

6.526.1 Detailed Description

Parallel implementation base for std::search() and std::search_n(). This file is a GNU parallel extension to the Standard C++ Library.

6.527 set File Reference

Macros

- `#define _GLIBCXX_SET`

6.527.1 Detailed Description

This is a Standard C++ Library header.

6.528 set File Reference

Macros

- `#define _GLIBCXX_DEBUG_SET`

6.528.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.529 set File Reference

Macros

- `#define _GLIBCXX_PROFILE_SET`

6.529.1 Detailed Description

This file is a GNU profile extension to the Standard C++ Library.

6.530 set File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_SET`

Typedefs

- `template<typename _Key, typename _Compare = less<_Key>>
using std::experimental::fundamentals_v2::pmr::multiset = std::multiset< _Key, _Compare, polymorphic_↵
allocator< _Key >>`
- `template<typename _Key, typename _Compare = less<_Key>>
using std::experimental::fundamentals_v2::pmr::set = std::set< _Key, _Compare, polymorphic_allocator< ↵
_Key >>`

Functions

- `template<typename _Key, typename _Compare, typename _Alloc, typename _Predicate >`
`void std::experimental::fundamentals_v2::erase_if (set< _Key, _Compare, _Alloc > &__cont, _Predicate _↔
_pred)`
- `template<typename _Key, typename _Compare, typename _Alloc, typename _Predicate >`
`void std::experimental::fundamentals_v2::erase_if (multiset< _Key, _Compare, _Alloc > &__cont, _Predicate
__pred)`

6.530.1 Detailed Description

This is a TS C++ Library header.

6.531 set.h File Reference

Classes

- class `std::__debug::set< _Key, _Compare, _Allocator >`

Namespaces

- `std`
- `std::__debug`

Functions

- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool std::__debug::operator!= (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare,
_Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool std::__debug::operator< (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare,
_Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool std::__debug::operator<= (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _↔
_Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool std::__debug::operator== (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare,
_Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool std::__debug::operator> (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare,
_Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`bool std::__debug::operator>= (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _↔
_Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >`
`void std::__debug::swap (set< _Key, _Compare, _Allocator > &__x, set< _Key, _Compare, _Allocator > &__y)`
`noexcept(/*conditional */)`

6.531.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.532 set.h File Reference

Classes

- class [std::__profile::set<_Key, _Compare, _Allocator>](#)

Namespaces

- [std](#)
- [std::__profile](#)

Functions

- `template<typename _Key, typename _Compare, typename _Allocator >
bool std::__profile::operator!= (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >
bool std::__profile::operator< (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >
bool std::__profile::operator<= (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >
bool std::__profile::operator== (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >
bool std::__profile::operator> (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >
bool std::__profile::operator>= (const set< _Key, _Compare, _Allocator > &__lhs, const set< _Key, _Compare, _Allocator > &__rhs)`
- `template<typename _Key, typename _Compare, typename _Allocator >
void std::__profile::swap (set< _Key, _Compare, _Allocator > &__x, set< _Key, _Compare, _Allocator > &__y)
noexcept(/*conditional */)`

6.532.1 Detailed Description

This file is a GNU profile extension to the Standard C++ Library.

6.533 set_operations.h File Reference

Namespaces

- [__gnu_parallel](#)

Functions

- `template<typename _Iter, typename _OutputIterator >`
`_OutputIterator __gnu_parallel::__copy_tail (std::pair< _Iter, _Iter > __b, std::pair< _Iter, _Iter > __e, _↵`
`_OutputIterator __r)`
- `template<typename _Iter, typename _OutputIterator, typename _Compare >`
`_OutputIterator __gnu_parallel::__parallel_set_difference (_Iter __begin1, _Iter __end1, _Iter __begin2, _↵`
`_Iter __end2, _OutputIterator __result, _Compare __comp)`
- `template<typename _Iter, typename _OutputIterator, typename _Compare >`
`_OutputIterator __gnu_parallel::__parallel_set_intersection (_Iter __begin1, _Iter __end1, _Iter __begin2,`
`_Iter __end2, _OutputIterator __result, _Compare __comp)`
- `template<typename _Iter, typename _OutputIterator, typename _Operation >`
`_OutputIterator __gnu_parallel::__parallel_set_operation (_Iter __begin1, _Iter __end1, _Iter __begin2, _↵`
`_Iter __end2, _OutputIterator __result, _Operation __op)`
- `template<typename _Iter, typename _OutputIterator, typename _Compare >`
`_OutputIterator __gnu_parallel::__parallel_set_symmetric_difference (_Iter __begin1, _Iter __end1, _Iter`
`__begin2, _Iter __end2, _OutputIterator __result, _Compare __comp)`
- `template<typename _Iter, typename _OutputIterator, typename _Compare >`
`_OutputIterator __gnu_parallel::__parallel_set_union (_Iter __begin1, _Iter __end1, _Iter __begin2, _Iter`
`__end2, _OutputIterator __result, _Compare __comp)`

6.533.1 Detailed Description

Parallel implementations of set operations for random-access iterators. This file is a GNU parallel extension to the Standard C++ Library.

6.534 settings.h File Reference

Classes

- `struct __gnu_parallel::Settings`

Namespaces

- `__gnu_parallel`

Macros

- `#define GLIBCXX_PARALLEL_CONDITION(__c)`

6.534.1 Detailed Description

Runtime settings and tuning parameters, heuristics to decide whether to use parallelized algorithms. This file is a GNU parallel extension to the Standard C++ Library.

6.534.2 parallelization_decision

The decision whether to run an algorithm in parallel.

There are several ways the user can switch on and __off the parallel execution of an algorithm, both at compile- and run-time.

Only sequential execution can be forced at compile-time. This reduces code size and protects code parts that have non-thread-safe side effects.

Ultimately, forcing parallel execution at compile-time makes sense. Often, the sequential algorithm implementation is used as a subroutine, so no reduction in code size can be achieved. Also, the machine the program is run on might have only one processor core, so to avoid overhead, the algorithm is executed sequentially.

To force sequential execution of an algorithm ultimately at compile-time, the user must add the tag `gnu_parallel↵::sequential_tag()` to the end of the parameter list, e. g.

```
std::sort(__v.begin(), __v.end(), __gnu_parallel::sequential_tag());
```

This is compatible with all overloaded algorithm variants. No additional code will be instantiated, at all. The same holds for most algorithm calls with iterators not providing random access.

If the algorithm call is not forced to be executed sequentially at compile-time, the decision is made at run-time. The global variable `__gnu_parallel::_Settings::algorithm_strategy` is checked. `_`It is a tristate variable corresponding to:

a. `force_sequential`, meaning the sequential algorithm is executed. b. `force_parallel`, meaning the parallel algorithm is executed. c. `heuristic`

For heuristic, the parallel algorithm implementation is called only if the input size is sufficiently large. For most algorithms, the input size is the (combined) length of the input sequence(`_s`). The threshold can be set by the user, individually for each algorithm. The according variables are called `gnu_parallel::_Settings::[algorithm]_minimal_n`.

For some of the algorithms, there are even more tuning options, e. g. the ability to choose from multiple algorithm variants. See below for details.

6.534.3 Macro Definition Documentation

6.534.3.1 `#define GLIBCXX_PARALLEL_CONDITION(__c)`

Determine at compile(?) -time if the parallel variant of an algorithm should be called.

Parameters

<code>__↵</code>	A condition that is convertible to bool that is overruled by <code>__gnu_parallel::_Settings::algorithm_strategy</code> .
<code>_c</code>	Usually a decision based on the input size.

Definition at line 95 of file settings.h.

Referenced by `__gnu_parallel::multiway_merge()`, and `__gnu_parallel::multiway_merge_sentinels()`.

6.535 `shared_mutex` File Reference

Classes

- class `std::__shared_mutex_cv`
- class `std::shared_lock<_Mutex>`
- class `std::shared_timed_mutex`

Namespaces

- `std`

Macros

- `#define _GLIBCXX_SHARED_MUTEX`
- `#define __cpp_lib_shared_timed_mutex`
- `using std::__shared_timed_mutex_base = __shared_mutex_cv`
- `template<typename _Mutex>`
`void std::swap(shared_lock<_Mutex> &__x, shared_lock<_Mutex> &__y) noexcept`

6.535.1 Detailed Description

This is a Standard C++ Library header.

6.536 `shared_ptr.h` File Reference

Classes

- class `std::enable_shared_from_this<_Tp>`
- struct `std::hash<shared_ptr<_Tp>>`
- struct `std::owner_less<_Tp>`
- struct `std::owner_less<shared_ptr<_Tp>>`
- struct `std::owner_less<weak_ptr<_Tp>>`
- class `std::shared_ptr<_Tp>`
- class `std::weak_ptr<_Tp>`

Namespaces

- `std`

Functions

- `template<typename _Tp1, typename _Tp2 >`
`void std::enable_shared_from_this_helper (const __shared_count<> &__pn, const enable_shared_from_←`
`__this< _Tp1 > * __pe, const _Tp2 * __px) noexcept`
- `template<typename _Tp, typename _Alloc, typename... _Args>`
`shared_ptr< _Tp > std::allocate_shared (const _Alloc &__a, _Args &&... __args)`
- `template<typename _Tp, typename _Tp1 >`
`shared_ptr< _Tp > std::const_pointer_cast (const shared_ptr< _Tp1 > &__r) noexcept`
- `template<typename _Tp, typename _Tp1 >`
`shared_ptr< _Tp > std::dynamic_pointer_cast (const shared_ptr< _Tp1 > &__r) noexcept`
- `template<typename _Del, typename _Tp, _Lock_policy _Lp>`
`_Del * std::get_deleter (const __shared_ptr< _Tp, _Lp > &__p) noexcept`
- `template<typename _Tp, typename... _Args>`
`shared_ptr< _Tp > std::make_shared (_Args &&... __args)`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::operator!= (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::operator!= (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool std::operator!= (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::operator< (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::operator< (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool std::operator< (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Ch, typename _Tr, typename _Tp, _Lock_policy _Lp>`
`std::basic_ostream< _Ch, _Tr > & std::operator<< (std::basic_ostream< _Ch, _Tr > &__os, const __shared_←`
`__ptr< _Tp, _Lp > &__p)`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::operator<= (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::operator<= (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool std::operator<= (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::operator== (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::operator== (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool std::operator== (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::operator> (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::operator> (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool std::operator> (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::operator>= (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::operator>= (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`

- `template<typename _Tp >`
`bool std::operator>= (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp, typename _Tp1 >`
`shared_ptr< _Tp > std::static_pointer_cast (const shared_ptr< _Tp1 > &__r) noexcept`
- `template<typename _Tp >`
`void std::swap (shared_ptr< _Tp > &__a, shared_ptr< _Tp > &__b) noexcept`
- `template<typename _Tp >`
`void std::swap (weak_ptr< _Tp > &__a, weak_ptr< _Tp > &__b) noexcept`

6.536.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

6.537 `shared_ptr.h` File Reference

Classes

- `struct std::experimental::fundamentals_v2::owner_less< shared_ptr< _Tp > >`
- `struct std::experimental::fundamentals_v2::owner_less< weak_ptr< _Tp > >`
- `struct std::hash< experimental::shared_ptr< _Tp > >`

Namespaces

- `std`

Macros

- `#define __cpp_lib_experimental_shared_ptr_arrays`

Typedefs

- `template<typename _Tp, _Lock_policy _Lp = __default_lock_policy>`
`using std::experimental::fundamentals_v2::__shared_ptr = std::__shared_ptr< __libfund_v1< _Tp >, _Lp >`
- `template<typename _Tp, _Lock_policy _Lp = __default_lock_policy>`
`using std::experimental::fundamentals_v2::__weak_ptr = std::__weak_ptr< __libfund_v1< _Tp >, _Lp >`

Functions

- `template<typename _Tp >`
`bool std::experimental::fundamentals_v2::atomic_compare_exchange_strong (shared_ptr< _Tp > *__p,`
`shared_ptr< _Tp > *__v, shared_ptr< _Tp > __w)`
- `template<typename _Tp >`
`bool std::experimental::fundamentals_v2::atomic_compare_exchange_strong_explicit (shared_ptr< _Tp`
`> *__p, shared_ptr< _Tp > *__v, shared_ptr< _Tp > __w, memory_order __success, memory_order __↵`
`failure)`
- `template<typename _Tp >`
`bool std::experimental::fundamentals_v2::atomic_compare_exchange_weak (shared_ptr< _Tp > *__↵`
`p, shared_ptr< _Tp > *__v, shared_ptr< _Tp > __w)`
- `template<typename _Tp >`
`bool std::experimental::fundamentals_v2::atomic_compare_exchange_weak_explicit (shared_ptr< _Tp >`
`*__p, shared_ptr< _Tp > *__v, shared_ptr< _Tp > __w, memory_order __success, memory_order __failure)`
- `template<typename _Tp >`
`void std::experimental::fundamentals_v2::atomic_exchange (shared_ptr< _Tp > *__p, shared_ptr< _Tp >`
`__r)`
- `template<typename _Tp >`
`shared_ptr< _Tp > std::experimental::fundamentals_v2::atomic_exchange_explicit (const shared_ptr< ↵`
`_Tp > *__p, shared_ptr< _Tp > __r, memory_order __mo)`
- `template<typename _Tp >`
`bool std::experimental::fundamentals_v2::atomic_is_lock_free (const shared_ptr< _Tp > *__p)`
- `template<typename _Tp >`
`shared_ptr< _Tp > std::experimental::fundamentals_v2::atomic_load (const shared_ptr< _Tp > *__p)`
- `template<typename _Tp >`
`shared_ptr< _Tp > std::experimental::fundamentals_v2::atomic_load_explicit (const shared_ptr< _Tp >`
`*__p, memory_order __mo)`
- `template<typename _Tp >`
`void std::experimental::fundamentals_v2::atomic_store (shared_ptr< _Tp > *__p, shared_ptr< _Tp > __r)`
- `template<typename _Tp >`
`shared_ptr< _Tp > std::experimental::fundamentals_v2::atomic_store_explicit (const shared_ptr< _Tp >`
`*__p, shared_ptr< _Tp > __r, memory_order __mo)`
- `template<typename _Tp, typename _Tp1 >`
`shared_ptr< _Tp > std::experimental::fundamentals_v2::const_pointer_cast (const shared_ptr< _Tp1 >`
`&__r) noexcept`
- `template<typename _Tp, typename _Tp1 >`
`shared_ptr< _Tp > std::experimental::fundamentals_v2::dynamic_pointer_cast (const shared_ptr< _Tp1`
`> &__r) noexcept`
- `template<typename _Del, typename _Tp, _Lock_policy _Lp>`
`_Del * std::experimental::fundamentals_v2::get_deleter (const __shared_ptr< _Tp, _Lp > &__p) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::experimental::fundamentals_v2::operator!= (const shared_ptr< _Tp1 > &__a, const shared_ptr<`
`_Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::experimental::fundamentals_v2::operator!= (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool std::experimental::fundamentals_v2::operator!= (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::experimental::fundamentals_v2::operator< (const shared_ptr< _Tp1 > &__a, const shared_ptr<`
`_Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::experimental::fundamentals_v2::operator< (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`

- `template<typename _Tp >`
`bool std::experimental::fundamentals_v2::operator< (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Ch, typename _Tr, typename _Tp, _Lock_policy _Lp>`
`std::basic_ostream< _Ch, _Tr > & std::experimental::fundamentals_v2::operator<< (std::basic_ostream< _Ch, _Tr > &__os, const __shared_ptr< _Tp, _Lp > &__p)`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::experimental::fundamentals_v2::operator<= (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::experimental::fundamentals_v2::operator<= (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool std::experimental::fundamentals_v2::operator<= (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::experimental::fundamentals_v2::operator== (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::experimental::fundamentals_v2::operator== (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool std::experimental::fundamentals_v2::operator== (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::experimental::fundamentals_v2::operator> (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::experimental::fundamentals_v2::operator> (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool std::experimental::fundamentals_v2::operator> (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp1, typename _Tp2 >`
`bool std::experimental::fundamentals_v2::operator>= (const shared_ptr< _Tp1 > &__a, const shared_ptr< _Tp2 > &__b) noexcept`
- `template<typename _Tp >`
`bool std::experimental::fundamentals_v2::operator>= (const shared_ptr< _Tp > &__a, nullptr_t) noexcept`
- `template<typename _Tp >`
`bool std::experimental::fundamentals_v2::operator>= (nullptr_t, const shared_ptr< _Tp > &__a) noexcept`
- `template<typename _Tp, typename _Tp1 >`
`shared_ptr< _Tp > std::experimental::fundamentals_v2::reinterpret_pointer_cast (const shared_ptr< _Tp1 > &__r) noexcept`
- `template<typename _Tp, typename _Tp1 >`
`shared_ptr< _Tp > std::experimental::fundamentals_v2::static_pointer_cast (const shared_ptr< _Tp1 > &__r) noexcept`
- `template<typename _Tp >`
`void std::experimental::fundamentals_v2::swap (shared_ptr< _Tp > &__a, shared_ptr< _Tp > &__b) noexcept`
- `template<typename _Tp >`
`void std::experimental::fundamentals_v2::swap (weak_ptr< _Tp > &__a, weak_ptr< _Tp > &__b) noexcept`

Variables

- `template<typename _Yp, typename _Tp >`
`constexpr bool std::__sp_compatible_v`
- `template<typename _Tp, typename _Yp >`
`constexpr bool std::__sp_is_constructible_v`

6.537.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<experimental/memory>`.

6.537.2 Function Documentation

6.537.2.1 `template<typename _Del, typename _Tp, _Lock_policy _Lp> _Del* std::experimental::fundamentals_v2::get_deleter (const __shared_ptr<_Tp, _Lp> &__p) [inline], [noexcept]`

C++14 §20.8.2.2.10.

Definition at line 1133 of file `experimental/bits/shared_ptr.h`.

6.538 shared_ptr_atomic.h File Reference

Namespaces

- [std](#)

Functions

- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::atomic_is_lock_free (const __shared_ptr<_Tp, _Lp> *__p)`
- `template<typename _Tp>`
`bool std::atomic_is_lock_free (const shared_ptr<_Tp> *__p)`
- `template<typename _Tp>`
`shared_ptr<_Tp> std::atomic_load_explicit (const shared_ptr<_Tp> *__p, memory_order)`
- `template<typename _Tp>`
`shared_ptr<_Tp> std::atomic_load (const shared_ptr<_Tp> *__p)`
- `template<typename _Tp, _Lock_policy _Lp>`
`__shared_ptr<_Tp, _Lp> std::atomic_load_explicit (const __shared_ptr<_Tp, _Lp> *__p, memory_order)`
- `template<typename _Tp, _Lock_policy _Lp>`
`__shared_ptr<_Tp, _Lp> std::atomic_load (const __shared_ptr<_Tp, _Lp> *__p)`
- `template<typename _Tp>`
`void std::atomic_store_explicit (shared_ptr<_Tp> *__p, shared_ptr<_Tp> __r, memory_order)`
- `template<typename _Tp>`
`void std::atomic_store (shared_ptr<_Tp> *__p, shared_ptr<_Tp> __r)`
- `template<typename _Tp, _Lock_policy _Lp>`
`void std::atomic_store_explicit (__shared_ptr<_Tp, _Lp> *__p, __shared_ptr<_Tp, _Lp> __r, memory_order)`
- `template<typename _Tp, _Lock_policy _Lp>`
`void std::atomic_store (__shared_ptr<_Tp, _Lp> *__p, __shared_ptr<_Tp, _Lp> __r)`

- `template<typename _Tp >`
`shared_ptr< _Tp > std::atomic_exchange_explicit (shared_ptr< _Tp > *__p, shared_ptr< _Tp > __r, memory_order)`
- `template<typename _Tp >`
`shared_ptr< _Tp > std::atomic_exchange (shared_ptr< _Tp > *__p, shared_ptr< _Tp > __r)`
- `template<typename _Tp, _Lock_policy _Lp>`
`__shared_ptr< _Tp, _Lp > std::atomic_exchange_explicit (__shared_ptr< _Tp, _Lp > *__p, __shared_ptr< _Tp, _Lp > __r, memory_order)`
- `template<typename _Tp, _Lock_policy _Lp>`
`__shared_ptr< _Tp, _Lp > std::atomic_exchange (__shared_ptr< _Tp, _Lp > *__p, __shared_ptr< _Tp, _Lp > __r)`

- `template<typename _Tp >`
`bool std::atomic_compare_exchange_strong_explicit (shared_ptr< _Tp > *__p, shared_ptr< _Tp > *__v, shared_ptr< _Tp > __w, memory_order, memory_order)`
- `template<typename _Tp >`
`bool std::atomic_compare_exchange_strong (shared_ptr< _Tp > *__p, shared_ptr< _Tp > *__v, shared_ptr< _Tp > __w)`
- `template<typename _Tp >`
`bool std::atomic_compare_exchange_weak_explicit (shared_ptr< _Tp > *__p, shared_ptr< _Tp > *__v, shared_ptr< _Tp > __w, memory_order __success, memory_order __failure)`
- `template<typename _Tp >`
`bool std::atomic_compare_exchange_weak (shared_ptr< _Tp > *__p, shared_ptr< _Tp > *__v, shared_ptr< _Tp > __w)`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::atomic_compare_exchange_strong_explicit (__shared_ptr< _Tp, _Lp > *__p, __shared_ptr< _Tp, _Lp > *__v, __shared_ptr< _Tp, _Lp > __w, memory_order, memory_order)`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::atomic_compare_exchange_strong (__shared_ptr< _Tp, _Lp > *__p, __shared_ptr< _Tp, _Lp > *__v, __shared_ptr< _Tp, _Lp > __w)`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::atomic_compare_exchange_weak_explicit (__shared_ptr< _Tp, _Lp > *__p, __shared_ptr< _Tp, _Lp > *__v, __shared_ptr< _Tp, _Lp > __w, memory_order __success, memory_order __failure)`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::atomic_compare_exchange_weak (__shared_ptr< _Tp, _Lp > *__p, __shared_ptr< _Tp, _Lp > *__v, __shared_ptr< _Tp, _Lp > __w)`

6.538.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

6.539 shared_ptr_base.h File Reference

Classes

- `struct std::_Sp_ebo_helper< _Nm, _Tp, false >`
- `struct std::_Sp_ebo_helper< _Nm, _Tp, true >`
- `class std::bad_weak_ptr`
- `class std::enable_shared_from_this< _Tp >`
- `struct std::hash< __shared_ptr< _Tp, _Lp > >`
- `struct std::owner_less< _Tp >`
- `class std::shared_ptr< _Tp >`
- `class std::weak_ptr< _Tp >`

Namespaces

- [std](#)

Functions

- `template<typename _Tp, _Lock_policy _Lp, typename _Alloc, typename... _Args>
__shared_ptr< _Tp, _Lp > std::allocate_shared (const _Alloc &__a, _Args &&...__args)`
- `template<typename _Tp1, typename _Tp2 >
void std::enable_shared_from_this_helper (const __shared_count<> &__pn, const enable_shared_from_←
_this< _Tp1 > *__pe, const _Tp2 *__px) noexcept`
- `template<_Lock_policy _Lp, typename _Tp1, typename _Tp2 >
void std::enable_shared_from_this_helper (const __shared_count< _Lp > &, const __enable_shared_←
from_this< _Tp1, _Lp > *, const _Tp2 *) noexcept`
- `template<_Lock_policy _Lp>
void std::enable_shared_from_this_helper (const __shared_count< _Lp > &,...) noexcept`
- `template<_Lock_policy _Lp1, typename _Tp1, typename _Tp2 >
void std::enable_shared_from_this_helper (const __shared_count< _Lp1 > &__pn, const __enable_←
shared_from_this< _Tp1, _Lp1 > *__pe, const _Tp2 *__px) noexcept`
- `template<typename _Tp, _Lock_policy _Lp, typename... _Args>
__shared_ptr< _Tp, _Lp > std::make_shared (_Args &&...__args)`
- `void std::throw_bad_weak_ptr ()`
- `template<typename _Tp, typename _Tp1, _Lock_policy _Lp>
__shared_ptr< _Tp, _Lp > std::const_pointer_cast (const __shared_ptr< _Tp1, _Lp > &__r) noexcept`
- `template<typename _Tp, typename _Tp1, _Lock_policy _Lp>
__shared_ptr< _Tp, _Lp > std::dynamic_pointer_cast (const __shared_ptr< _Tp1, _Lp > &__r) noexcept`
- `template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>
bool std::operator!= (const __shared_ptr< _Tp1, _Lp > &__a, const __shared_ptr< _Tp2, _Lp > &__b) noex-
cept`
- `template<typename _Tp, _Lock_policy _Lp>
bool std::operator!= (const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>
bool std::operator!= (nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`
- `template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>
bool std::operator< (const __shared_ptr< _Tp1, _Lp > &__a, const __shared_ptr< _Tp2, _Lp > &__b) noex-
cept`
- `template<typename _Tp, _Lock_policy _Lp>
bool std::operator< (const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>
bool std::operator< (nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`
- `template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>
bool std::operator<= (const __shared_ptr< _Tp1, _Lp > &__a, const __shared_ptr< _Tp2, _Lp > &__b) noex-
cept`
- `template<typename _Tp, _Lock_policy _Lp>
bool std::operator<= (const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>
bool std::operator<= (nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`
- `template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>
bool std::operator== (const __shared_ptr< _Tp1, _Lp > &__a, const __shared_ptr< _Tp2, _Lp > &__b) noex-
cept`
- `template<typename _Tp, _Lock_policy _Lp>
bool std::operator== (const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`

- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::operator== (nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`
- `template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>`
`bool std::operator> (const __shared_ptr< _Tp1, _Lp > &__a, const __shared_ptr< _Tp2, _Lp > &__b) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::operator> (const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::operator> (nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`
- `template<typename _Tp1, typename _Tp2, _Lock_policy _Lp>`
`bool std::operator>= (const __shared_ptr< _Tp1, _Lp > &__a, const __shared_ptr< _Tp2, _Lp > &__b) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::operator>= (const __shared_ptr< _Tp, _Lp > &__a, nullptr_t) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`bool std::operator>= (nullptr_t, const __shared_ptr< _Tp, _Lp > &__a) noexcept`
- `template<typename _Tp, typename _Tp1, _Lock_policy _Lp>`
`__shared_ptr< _Tp, _Lp > std::static_pointer_cast (const __shared_ptr< _Tp1, _Lp > &__r) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`void std::swap (__shared_ptr< _Tp, _Lp > &__a, __shared_ptr< _Tp, _Lp > &__b) noexcept`
- `template<typename _Tp, _Lock_policy _Lp>`
`void std::swap (__weak_ptr< _Tp, _Lp > &__a, __weak_ptr< _Tp, _Lp > &__b) noexcept`

6.539.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

6.540 `size_fn_imps.hpp` File Reference

6.540.1 Detailed Description

Contains implementations of `cc_ht_map_'s` entire container size related functions.

6.541 `slice_array.h` File Reference

Classes

- class `std::slice`
- class `std::slice_array< _Tp >`

Namespaces

- `std`

Macros

- `#define _DEFINE_VALARRAY_OPERATOR(_Op, _Name)`

6.541.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<valarray>`.

6.542 slist File Reference

Classes

- class [__gnu_cxx::slist< _Tp, _Alloc >](#)

Namespaces

- [__gnu_cxx](#)
- [std](#)

Macros

- `#define _SLIST`

Functions

- `_Slist_node_base * __gnu_cxx::__slist_make_link (_Slist_node_base * __prev_node, _Slist_node_base * __new_node)`
- `_Slist_node_base * __gnu_cxx::__slist_previous (_Slist_node_base * __head, const _Slist_node_base * __node)`
- `const _Slist_node_base * __gnu_cxx::__slist_previous (const _Slist_node_base * __head, const _Slist_node_base * __node)`
- `_Slist_node_base * __gnu_cxx::__slist_reverse (_Slist_node_base * __node)`
- `size_t __gnu_cxx::__slist_size (_Slist_node_base * __node)`
- `void __gnu_cxx::__slist_splice_after (_Slist_node_base * __pos, _Slist_node_base * __before_first, _Slist_node_base * __before_last)`
- `void __gnu_cxx::__slist_splice_after (_Slist_node_base * __pos, _Slist_node_base * __head)`
- `template<class _Tp, class _Alloc>
bool __gnu_cxx::operator!= (const slist< _Tp, _Alloc > &_SL1, const slist< _Tp, _Alloc > &_SL2)`
- `template<class _Tp, class _Alloc>
bool __gnu_cxx::operator< (const slist< _Tp, _Alloc > &_SL1, const slist< _Tp, _Alloc > &_SL2)`
- `template<class _Tp, class _Alloc>
bool __gnu_cxx::operator<= (const slist< _Tp, _Alloc > &_SL1, const slist< _Tp, _Alloc > &_SL2)`
- `template<class _Tp, class _Alloc>
bool __gnu_cxx::operator== (const slist< _Tp, _Alloc > &_SL1, const slist< _Tp, _Alloc > &_SL2)`
- `template<class _Tp, class _Alloc>
bool __gnu_cxx::operator> (const slist< _Tp, _Alloc > &_SL1, const slist< _Tp, _Alloc > &_SL2)`
- `template<class _Tp, class _Alloc>
bool __gnu_cxx::operator>= (const slist< _Tp, _Alloc > &_SL1, const slist< _Tp, _Alloc > &_SL2)`
- `template<class _Tp, class _Alloc>
void __gnu_cxx::swap (slist< _Tp, _Alloc > &__x, slist< _Tp, _Alloc > &__y)`

6.542.1 Detailed Description

This file is a GNU extension to the Standard C++ Library (possibly containing extensions from the HP/SGI STL subset).

6.543 sort.h File Reference

Namespaces

- [__gnu_parallel](#)

Functions

- `template<bool __stable, typename _RAIter, typename _Compare, typename _Parallelism >`
`void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, _Parallelism __↵`
`parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, multiway_mergesort↵`
`__tag __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, multiway_mergesort↵`
`__exact_tag __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, multiway_mergesort↵`
`__sampling_tag __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, quicksort_tag __↵`
`parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, balanced_quicksort↵`
`__tag __parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, default_parallel_tag`
`__parallelism)`
- `template<bool __stable, typename _RAIter, typename _Compare >`
`void __gnu_parallel::__parallel_sort (_RAIter __begin, _RAIter __end, _Compare __comp, parallel_tag __↵`
`parallelism)`

6.543.1 Detailed Description

Parallel sorting algorithm switch. This file is a GNU parallel extension to the Standard C++ Library.

6.544 specfun.h File Reference

Namespaces

- [__gnu_cxx](#)
- [std](#)

Macros

- `#define __cpp_lib_math_special_functions`
- `#define __STDCPP_MATH_SPEC_FUNCS__`

Functions

- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::assoc_laguerre (unsigned int __n, unsigned int __m, _Tp __x)`
- `float std::assoc_laguerref (unsigned int __n, unsigned int __m, float __x)`
- `long double std::assoc_laguerrel (unsigned int __n, unsigned int __m, long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::assoc_legendre (unsigned int __l, unsigned int __m, _Tp __x)`
- `float std::assoc_legendref (unsigned int __l, unsigned int __m, float __x)`
- `long double std::assoc_legendrel (unsigned int __l, unsigned int __m, long double __x)`
- `template<typename _Tpa, typename _Tpb >`
`__gnu_cxx::__promote_2< _Tpa, _Tpb >::__type std::beta (_Tpa __a, _Tpb __b)`
- `float std::betaf (float __a, float __b)`
- `long double std::betal (long double __a, long double __b)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::comp_ellint_1 (_Tp __k)`
- `float std::comp_ellint_1f (float __k)`
- `long double std::comp_ellint_1l (long double __k)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::comp_ellint_2 (_Tp __k)`
- `float std::comp_ellint_2f (float __k)`
- `long double std::comp_ellint_2l (long double __k)`
- `template<typename _Tp, typename _Tpn >`
`__gnu_cxx::__promote_2< _Tp, _Tpn >::__type std::comp_ellint_3 (_Tp __k, _Tpn __nu)`
- `float std::comp_ellint_3f (float __k, float __nu)`
- `long double std::comp_ellint_3l (long double __k, long double __nu)`
- `template<typename _Tpa, typename _Tpc, typename _Tp >`
`__gnu_cxx::__promote_3< _Tpa, _Tpc, _Tp >::__type __gnu_cxx::conf_hyperg (_Tpa __a, _Tpc __c, _Tp __x)`
- `float __gnu_cxx::conf_hypergf (float __a, float __c, float __x)`
- `long double __gnu_cxx::conf_hypergl (long double __a, long double __c, long double __x)`
- `template<typename _Tpnu, typename _Tp >`
`__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::cyl_bessel_i (_Tpnu __nu, _Tp __x)`
- `float std::cyl_bessel_if (float __nu, float __x)`
- `long double std::cyl_bessel_il (long double __nu, long double __x)`
- `template<typename _Tpnu, typename _Tp >`
`__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::cyl_bessel_j (_Tpnu __nu, _Tp __x)`
- `float std::cyl_bessel_jf (float __nu, float __x)`
- `long double std::cyl_bessel_jl (long double __nu, long double __x)`
- `template<typename _Tpnu, typename _Tp >`
`__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::cyl_bessel_k (_Tpnu __nu, _Tp __x)`
- `float std::cyl_bessel_kf (float __nu, float __x)`
- `long double std::cyl_bessel_kl (long double __nu, long double __x)`
- `template<typename _Tpnu, typename _Tp >`
`__gnu_cxx::__promote_2< _Tpnu, _Tp >::__type std::cyl_neumann (_Tpnu __nu, _Tp __x)`
- `float std::cyl_neumannf (float __nu, float __x)`
- `long double std::cyl_neumannl (long double __nu, long double __x)`

- `template<typename _Tp, typename _Tpp >`
`__gnu_cxx::__promote_2< _Tp, _Tpp >::__type std::ellint_1 (_Tp __k, _Tpp __phi)`
- `float std::ellint_1f (float __k, float __phi)`
- `long double std::ellint_1l (long double __k, long double __phi)`
- `template<typename _Tp, typename _Tpp >`
`__gnu_cxx::__promote_2< _Tp, _Tpp >::__type std::ellint_2 (_Tp __k, _Tpp __phi)`
- `float std::ellint_2f (float __k, float __phi)`
- `long double std::ellint_2l (long double __k, long double __phi)`
- `template<typename _Tp, typename _Tpn, typename _Tpp >`
`__gnu_cxx::__promote_3< _Tp, _Tpn, _Tpp >::__type std::ellint_3 (_Tp __k, _Tpn __nu, _Tpp __phi)`
- `float std::ellint_3f (float __k, float __nu, float __phi)`
- `long double std::ellint_3l (long double __k, long double __nu, long double __phi)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::expint (_Tp __x)`
- `float std::expintf (float __x)`
- `long double std::expintl (long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::hermite (unsigned int __n, _Tp __x)`
- `float std::hermitef (unsigned int __n, float __x)`
- `long double std::hermitel (unsigned int __n, long double __x)`
- `template<typename _Tpa, typename _Tpb, typename _Tpc, typename _Tp >`
`__gnu_cxx::__promote_4< _Tpa, _Tpb, _Tpc, _Tp >::__type __gnu_cxx::hyperg (_Tpa __a, _Tpb __b, _Tpc __c, _Tp __x)`
- `float __gnu_cxx::hypergf (float __a, float __b, float __c, float __x)`
- `long double __gnu_cxx::hypergl (long double __a, long double __b, long double __c, long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::laguerre (unsigned int __n, _Tp __x)`
- `float std::laguerref (unsigned int __n, float __x)`
- `long double std::laguerrel (unsigned int __n, long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::legendre (unsigned int __l, _Tp __x)`
- `float std::legendref (unsigned int __l, float __x)`
- `long double std::legendrel (unsigned int __l, long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::riemann_zeta (_Tp __s)`
- `float std::riemann_zetaf (float __s)`
- `long double std::riemann_zetal (long double __s)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::sph_bessel (unsigned int __n, _Tp __x)`
- `float std::sph_besself (unsigned int __n, float __x)`
- `long double std::sph_bessell (unsigned int __n, long double __x)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::sph_legendre (unsigned int __l, unsigned int __m, _Tp __theta)`
- `float std::sph_legendref (unsigned int __l, unsigned int __m, float __theta)`
- `long double std::sph_legendrel (unsigned int __l, unsigned int __m, long double __theta)`
- `template<typename _Tp >`
`__gnu_cxx::__promote< _Tp >::__type std::sph_neumann (unsigned int __n, _Tp __x)`
- `float std::sph_neumannf (unsigned int __n, float __x)`
- `long double std::sph_neumannl (unsigned int __n, long double __x)`

6.544.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<cmath>`.

6.545 `splay_fn_imps.hpp` File Reference

6.545.1 Detailed Description

Contains an implementation class for `splay_tree_`.

6.546 `splay_tree_.hpp` File Reference

Classes

- class [__gnu_pbds::detail::splay_tree_map< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_ASSERT_BASE_NODE_CONSISTENT(_Node)`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_S_TREE_BASE`
- `#define PB_DS_S_TREE_BASE_NAME`
- `#define PB_DS_S_TREE_NAME`

6.546.1 Detailed Description

Contains an implementation class for splay trees.

6.547 `split_fn_imps.hpp` File Reference

6.547.1 Detailed Description

Contains an implementation class for `pat_trie`.

6.548 split_join_fn_imps.hpp File Reference**6.548.1 Detailed Description**

Contains an implementation class for a binary_heap.

6.549 split_join_fn_imps.hpp File Reference**6.549.1 Detailed Description**

Contains an implementation class for a base of binomial heaps.

6.550 split_join_fn_imps.hpp File Reference**6.550.1 Detailed Description**

Contains an implementation class for bin_search_tree_.

6.551 split_join_fn_imps.hpp File Reference**6.551.1 Detailed Description**

Contains an implementation class for ov_tree_.

6.552 split_join_fn_imps.hpp File Reference**6.552.1 Detailed Description**

Contains an implementation class for a pairing heap.

6.553 split_join_fn_imps.hpp File Reference**6.553.1 Detailed Description**

Contains an implementation for rb_tree_.

6.554 `split_join_fn_imps.hpp` File Reference

6.554.1 Detailed Description

Contains an implementation for `rc_binomial_heap_`.

6.555 `split_join_fn_imps.hpp` File Reference

6.555.1 Detailed Description

Contains an implementation class for `splay_tree_`.

6.556 `split_join_fn_imps.hpp` File Reference

6.556.1 Detailed Description

Contains an implementation for `thin_heap_`.

6.557 `sso_string_base.h` File Reference

Namespaces

- [__gnu_cxx](#)

6.557.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ext/vstring.h>`.

6.558 `sstream` File Reference

Classes

- class [std::basic_istream<_CharT, _Traits, _Alloc>](#)
- class [std::basic_ostringstream<_CharT, _Traits, _Alloc>](#)
- class [std::basic_stringbuf<_CharT, _Traits, _Alloc>](#)
- class [std::basic_stringstream<_CharT, _Traits, _Alloc>](#)

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_SSTREAM`

Functions

- `template<class _CharT, class _Traits, class _Allocator >`
`void std::swap (basic_stringbuf< _CharT, _Traits, _Allocator > &__x, basic_stringbuf< _CharT, _Traits, _Allocator > &__y)`
- `template<class _CharT, class _Traits, class _Allocator >`
`void std::swap (basic_istream< _CharT, _Traits, _Allocator > &__x, basic_istream< _CharT, _Traits, _Allocator > &__y)`
- `template<class _CharT, class _Traits, class _Allocator >`
`void std::swap (basic_ostringstream< _CharT, _Traits, _Allocator > &__x, basic_ostringstream< _CharT, _Traits, _Allocator > &__y)`
- `template<class _CharT, class _Traits, class _Allocator >`
`void std::swap (basic_stringstream< _CharT, _Traits, _Allocator > &__x, basic_stringstream< _CharT, _Traits, _Allocator > &__y)`

6.558.1 Detailed Description

This is a Standard C++ Library header.

6.559 sstream.tcc File Reference

Namespaces

- [std](#)

Macros

- `#define _SSTREAM_TCC`

6.559.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<sstream>`.

6.560 stack File Reference

Macros

- `#define _GLIBCXX_STACK`

6.560.1 Detailed Description

This is a Standard C++ Library header.

6.561 standard_policies.hpp File Reference

Classes

- struct [__gnu_pbds::detail::default_comb_hash_fn](#)
- struct [__gnu_pbds::detail::default_eq_fn< Key >](#)
- struct [__gnu_pbds::detail::default_hash_fn< Key >](#)
- struct [__gnu_pbds::detail::default_probe_fn< Comb_Probe_Fn >](#)
- struct [__gnu_pbds::detail::default_resize_policy< Comb_Hash_Fn >](#)
- struct [__gnu_pbds::detail::default_trie_access_traits< Key >](#)
- struct [__gnu_pbds::detail::default_trie_access_traits< std::basic_string< Char, Char_Traits, std::allocator< char > > >](#)
- struct [__gnu_pbds::detail::default_update_policy](#)

Namespaces

- [__gnu_pbds](#)

Macros

- `#define __dtrie_alloc`
- `#define __dtrie_string`

Enumerations

- enum { **default_store_hash** }

6.561.1 Detailed Description

Contains standard policies for containers.

6.561.2 Enumeration Type Documentation

6.561.2.1 anonymous enum

Enumeration for default behavior of stored hash data.

Definition at line 74 of file standard_policies.hpp.

6.562 std_mutex.h File Reference

Classes

- struct [std::adopt_lock_t](#)
- struct [std::defer_lock_t](#)
- class [std::lock_guard< _Mutex >](#)
- class [std::mutex](#)
- struct [std::try_to_lock_t](#)
- class [std::unique_lock< _Mutex >](#)

Namespaces

- [std](#)

Functions

- [template<typename _Mutex >](#)
void [std::swap](#) ([unique_lock< _Mutex >](#) &__x, [unique_lock< _Mutex >](#) &__y) noexcept

Variables

- constexpr [adopt_lock_t](#) [std::adopt_lock](#)
- constexpr [defer_lock_t](#) [std::defer_lock](#)
- constexpr [try_to_lock_t](#) [std::try_to_lock](#)

6.562.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<mutex>`.

6.563 stdc++.h File Reference

6.563.1 Detailed Description

This is an implementation file for a precompiled header.

6.564 stdexcept File Reference

Classes

- class [std::domain_error](#)
- class [std::invalid_argument](#)
- class [std::length_error](#)
- class [std::logic_error](#)
- class [std::out_of_range](#)
- class [std::overflow_error](#)
- class [std::range_error](#)
- class [std::runtime_error](#)
- class [std::underflow_error](#)

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_STDEXCEPT`

Typedefs

- typedef basic_string< char > **std::__cow_string**
- typedef basic_string< char > **std::__sso_string**

6.564.1 Detailed Description

This is a Standard C++ Library header.

6.565 stdio_filebuf.h File Reference

Classes

- class [__gnu_cxx::stdio_filebuf< _CharT, _Traits >](#)

Namespaces

- [__gnu_cxx](#)

6.565.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.566 `stdio_sync_filebuf.h` File Reference

Classes

- class [__gnu_cxx::stdio_sync_filebuf<_CharT, _Traits>](#)

Namespaces

- [__gnu_cxx](#)

6.566.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.567 `stdlib.h` File Reference

6.567.1 Detailed Description

This is a Standard C++ Library header.

6.568 `stdtr1c++.h` File Reference

6.568.1 Detailed Description

This is an implementation file for a precompiled header.

6.569 `stl_algo.h` File Reference

Namespaces

- [std](#)

Enumerations

- enum { **_S_threshold** }
- enum { **_S_chunk_size** }

Functions

- `template<typename _ForwardIterator, typename _BinaryPredicate >`
`_ForwardIterator std::adjacent_find (_ForwardIterator __first, _ForwardIterator __last, _BinaryPredicate __↵`
`binary_pred)`
- `template<typename _RandomAccessIterator, typename _Distance, typename _Compare >`
`void std::chunk_insertion_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Distance`
`__chunk_size, _Compare __comp)`
- `template<typename _InputIterator, typename _Size, typename _OutputIterator >`
`_OutputIterator std::copy_n (_InputIterator __first, _Size __n, _OutputIterator __result, input_iterator_tag)`
- `template<typename _RandomAccessIterator, typename _Size, typename _OutputIterator >`
`_OutputIterator std::copy_n (_RandomAccessIterator __first, _Size __n, _OutputIterator __result, random_↵`
`access_iterator_tag)`
- `template<typename _InputIterator, typename _Predicate >`
`iterator_traits< _InputIterator >::difference_type std::count_if (_InputIterator __first, _InputIterator __last, ↵`
`_Predicate __pred)`
- `template<typename _ForwardIterator, typename _Tp, typename _CompareItTp, typename _CompareTpIt >`
`pair< _ForwardIterator, _ForwardIterator > std::equal_range (_ForwardIterator __first, _ForwardIterator __↵`
`last, const _Tp & __val, _CompareItTp __comp_it_val, _CompareTpIt __comp_val_it)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::final_insertion_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __↵`
`comp)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`_ForwardIterator1 std::find_end (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 ↵`
`__first2, _ForwardIterator2 __last2, forward_iterator_tag, forward_iterator_tag, _BinaryPredicate __comp)`
- `template<typename _BidirectionalIterator1, typename _BidirectionalIterator2, typename _BinaryPredicate >`
`_BidirectionalIterator1 std::find_end (_BidirectionalIterator1 __first1, _BidirectionalIterator1 __last1, ↵`
`_BidirectionalIterator2 __first2, _BidirectionalIterator2 __last2, bidirectional_iterator_tag, bidirectional_iterator_tag,`
`_BinaryPredicate __comp)`
- `template<typename _InputIterator, typename _Predicate >`
`_InputIterator std::find_if (_InputIterator __first, _InputIterator __last, _Predicate __pred, input_iterator_tag)`
- `template<typename _RandomAccessIterator, typename _Predicate >`
`_RandomAccessIterator std::find_if (_RandomAccessIterator __first, _RandomAccessIterator __last, ↵`
`_Predicate __pred, random_access_iterator_tag)`
- `template<typename _Iterator, typename _Predicate >`
`_Iterator std::find_if (_Iterator __first, _Iterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _Predicate >`
`_InputIterator std::find_if_not (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _Predicate, typename _Distance >`
`_InputIterator std::find_if_not_n (_InputIterator __first, _Distance & __len, _Predicate __pred)`
- `template<typename _EuclideanRingElement >`
`_EuclideanRingElement std::gcd (_EuclideanRingElement __m, _EuclideanRingElement __n)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::heap_select (_RandomAccessIterator __first, _RandomAccessIterator __middle, _RandomAccess↵`
`Iterator __last, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Compare >`
`bool std::includes (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2`
`__last2, _Compare __comp)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`void std::inplace_merge (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator`
`__last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::inplace_stable_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __↵`
`comp)`

- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::__insertion_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Size, typename _Compare >`
`void std::__introsort (_RandomAccessIterator __first, _RandomAccessIterator __nth, _RandomAccessIterator __last, _Size __depth_limit, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Size, typename _Compare >`
`void std::__introsort_loop (_RandomAccessIterator __first, _RandomAccessIterator __last, _Size __depth_limit, _Compare __comp)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`bool std::__is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _BinaryPredicate __pred)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`bool std::__is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2 __last2, _BinaryPredicate __pred)`
- `template<typename _ForwardIterator, typename _Compare >`
`_ForwardIterator std::__is_sorted_until (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Compare >`
`_GLIBCXX14_CONSTEXPR _ForwardIterator std::__max_element (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::__merge (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _BidirectionalIterator, typename _Distance, typename _Pointer, typename _Compare >`
`void std::__merge_adaptive (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last, _Distance __len1, _Distance __len2, _Pointer __buffer, _Distance __buffer_size, _Compare __comp)`
- `template<typename _RandomAccessIterator1, typename _RandomAccessIterator2, typename _Distance, typename _Compare >`
`void std::__merge_sort_loop (_RandomAccessIterator1 __first, _RandomAccessIterator1 __last, _RandomAccessIterator2 __result, _Distance __step_size, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Pointer, typename _Compare >`
`void std::__merge_sort_with_buffer (_RandomAccessIterator __first, _RandomAccessIterator __last, _Pointer __buffer, _Compare __comp)`
- `template<typename _BidirectionalIterator, typename _Distance, typename _Compare >`
`void std::__merge_without_buffer (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last, _Distance __len1, _Distance __len2, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Compare >`
`_GLIBCXX14_CONSTEXPR _ForwardIterator std::__min_element (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Compare >`
`_GLIBCXX14_CONSTEXPR pair<_ForwardIterator, _ForwardIterator > std::__minmax_element (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _Iterator, typename _Compare >`
`void std::__move_median_to_first (_Iterator __result, _Iterator __a, _Iterator __b, _Iterator __c, _Compare __comp)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::__move_merge (_InputIterator __first1, _InputIterator __last1, _InputIterator __first2, _InputIterator __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`void std::__move_merge_adaptive (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _BidirectionalIterator1, typename _BidirectionalIterator2, typename _BidirectionalIterator3, typename _Compare >`
`void std::__move_merge_adaptive_backward (_BidirectionalIterator1 __first1, _BidirectionalIterator1 __last1, _BidirectionalIterator2 __first2, _BidirectionalIterator2 __last2, _BidirectionalIterator3 __result, _Compare __comp)`

- `template<typename _BidirectionalIterator, typename _Compare >`
`bool std::next_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::partial_sort (_RandomAccessIterator __first, _RandomAccessIterator __middle, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _Compare >`
`_RandomAccessIterator std::partial_sort_copy (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __result_first, _RandomAccessIterator __result_last, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator std::partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred, forward_iterator_tag)`
- `template<typename _BidirectionalIterator, typename _Predicate >`
`_BidirectionalIterator std::partition (_BidirectionalIterator __first, _BidirectionalIterator __last, _Predicate __pred, bidirectional_iterator_tag)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`bool std::prev_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare __comp)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::remove_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Predicate __pred)`
- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator std::remove_if (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate, typename _Tp >`
`_OutputIterator std::replace_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Predicate __pred, const _Tp & __new_value)`
- `template<typename _BidirectionalIterator >`
`void std::reverse (_BidirectionalIterator __first, _BidirectionalIterator __last, bidirectional_iterator_tag)`
- `template<typename _RandomAccessIterator >`
`void std::reverse (_RandomAccessIterator __first, _RandomAccessIterator __last, random_access_iterator_tag)`
- `template<typename _ForwardIterator >`
`_ForwardIterator std::V2::rotate (_ForwardIterator __first, _ForwardIterator __middle, _ForwardIterator __last, forward_iterator_tag)`
- `template<typename _BidirectionalIterator >`
`_BidirectionalIterator std::V2::rotate (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last, bidirectional_iterator_tag)`
- `template<typename _RandomAccessIterator >`
`_RandomAccessIterator std::V2::rotate (_RandomAccessIterator __first, _RandomAccessIterator __middle, _RandomAccessIterator __last, random_access_iterator_tag)`
- `template<typename _BidirectionalIterator1, typename _BidirectionalIterator2, typename _Distance >`
`_BidirectionalIterator1 std::rotate_adaptive (_BidirectionalIterator1 __first, _BidirectionalIterator1 __middle, _BidirectionalIterator1 __last, _Distance __len1, _Distance __len2, _BidirectionalIterator2 __buffer, _Distance __buffer_size)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`_ForwardIterator1 std::search (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, _ForwardIterator2 __last2, _BinaryPredicate __predicate)`
- `template<typename _ForwardIterator, typename _Integer, typename _UnaryPredicate >`
`_ForwardIterator std::search_n (_ForwardIterator __first, _ForwardIterator __last, _Integer __count, _UnaryPredicate __unary_pred)`
- `template<typename _ForwardIterator, typename _Integer, typename _UnaryPredicate >`
`_ForwardIterator std::search_n_aux (_ForwardIterator __first, _ForwardIterator __last, _Integer __count, _UnaryPredicate __unary_pred, std::forward_iterator_tag)`
- `template<typename _RandomAccessIterator, typename _Integer, typename _UnaryPredicate >`
`_RandomAccessIterator std::search_n_aux (_RandomAccessIterator __first, _RandomAccessIterator __last, _Integer __count, _UnaryPredicate __unary_pred, std::random_access_iterator_tag)`

- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::set_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2,`
`_InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::set_intersection (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2,`
`_InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::set_symmetric_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2,`
`_InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::set_union (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2,`
`_OutputIterator __result, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator std::stable_partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _ForwardIterator, typename _Pointer, typename _Predicate, typename _Distance >`
`_ForwardIterator std::stable_partition_adaptive (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred,`
`_Distance __len, _Pointer __buffer, _Distance __buffer_size)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::stable_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Pointer, typename _Distance, typename _Compare >`
`void std::stable_sort_adaptive (_RandomAccessIterator __first, _RandomAccessIterator __last, _Pointer __buffer,`
`_Distance __buffer_size, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::unguarded_insertion_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::unguarded_linear_insert (_RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`_RandomAccessIterator std::unguarded_partition (_RandomAccessIterator __first, _RandomAccessIterator __last, _RandomAccessIterator __pivot,`
`_Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`_RandomAccessIterator std::unguarded_partition_pivot (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _ForwardIterator, typename _BinaryPredicate >`
`_ForwardIterator std::unique (_ForwardIterator __first, _ForwardIterator __last, _BinaryPredicate __binary_pred)`
- `template<typename _ForwardIterator, typename _OutputIterator, typename _BinaryPredicate >`
`_OutputIterator std::unique_copy (_ForwardIterator __first, _ForwardIterator __last, _OutputIterator __result, _BinaryPredicate __binary_pred,`
`forward_iterator_tag, output_iterator_tag)`
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryPredicate >`
`_OutputIterator std::unique_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _BinaryPredicate __binary_pred,`
`input_iterator_tag, output_iterator_tag)`
- `template<typename _InputIterator, typename _ForwardIterator, typename _BinaryPredicate >`
`_ForwardIterator std::unique_copy (_InputIterator __first, _InputIterator __last, _ForwardIterator __result, _BinaryPredicate __binary_pred,`
`input_iterator_tag, forward_iterator_tag)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`_ForwardIterator std::upper_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)`
- `template<typename _ForwardIterator >`
`_ForwardIterator std::adjacent_find (_ForwardIterator __first, _ForwardIterator __last)`

- `template<typename _ForwardIterator, typename _BinaryPredicate >`
`_ForwardIterator std::adjacent_find (_ForwardIterator __first, _ForwardIterator __last, _BinaryPredicate __↵`
`binary_pred)`
- `template<typename _InputIterator, typename _Predicate >`
`bool std::all_of (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _Predicate >`
`bool std::any_of (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _ForwardIterator, typename _Tp >`
`bool std::binary_search (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`bool std::binary_search (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Predicate __↵`
`pred)`
- `template<typename _InputIterator, typename _Size, typename _OutputIterator >`
`_OutputIterator std::copy_n (_InputIterator __first, _Size __n, _OutputIterator __result)`
- `template<typename _InputIterator, typename _Tp >`
`iterator_traits< _InputIterator >::difference_type std::count (_InputIterator __first, _InputIterator __last, const _Tp`
`&__value)`
- `template<typename _InputIterator, typename _Predicate >`
`iterator_traits< _InputIterator >::difference_type std::count_if (_InputIterator __first, _InputIterator __last, __↵`
`Predicate __pred)`
- `template<typename _ForwardIterator, typename _Tp >`
`pair< _ForwardIterator, _ForwardIterator > std::equal_range (_ForwardIterator __first, _ForwardIterator __last,`
`const _Tp &__val)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`pair< _ForwardIterator, _ForwardIterator > std::equal_range (_ForwardIterator __first, _ForwardIterator __last,`
`const _Tp &__val, _Compare __comp)`
- `template<typename _InputIterator, typename _Tp >`
`_InputIterator std::find (_InputIterator __first, _InputIterator __last, const _Tp &__val)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`_ForwardIterator1 std::find_end (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __↵`
`first2, _ForwardIterator2 __last2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`_ForwardIterator1 std::find_end (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __↵`
`first2, _ForwardIterator2 __last2, _BinaryPredicate __comp)`
- `template<typename _InputIterator, typename _ForwardIterator >`
`_InputIterator std::find_first_of (_InputIterator __first1, _InputIterator __last1, _ForwardIterator __first2, __↵`
`ForwardIterator __last2)`
- `template<typename _InputIterator, typename _ForwardIterator, typename _BinaryPredicate >`
`_InputIterator std::find_first_of (_InputIterator __first1, _InputIterator __last1, _ForwardIterator __first2, __↵`
`ForwardIterator __last2, _BinaryPredicate __comp)`
- `template<typename _InputIterator, typename _Predicate >`
`_InputIterator std::find_if (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _Predicate >`
`_InputIterator std::find_if_not (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _InputIterator, typename _Function >`
`_Function std::for_each (_InputIterator __first, _InputIterator __last, _Function __f)`
- `template<typename _ForwardIterator, typename _Generator >`
`void std::generate (_ForwardIterator __first, _ForwardIterator __last, _Generator __gen)`
- `template<typename _OutputIterator, typename _Size, typename _Generator >`
`_OutputIterator std::generate_n (_OutputIterator __first, _Size __n, _Generator __gen)`

- `template<typename _InputIterator1, typename _InputIterator2 >`
`bool std::includes (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __↵`
`last2)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _Compare >`
`bool std::includes (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __↵`
`last2, _Compare __comp)`
- `template<typename _BidirectionalIterator >`
`void std::inplace_merge (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __↵`
`last)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`void std::inplace_merge (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __↵`
`last, _Compare __comp)`
- `template<typename _InputIterator, typename _Predicate >`
`bool std::is_partitioned (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, __↵`
`BinaryPredicate __pred)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, __↵`
`ForwardIterator2 __last2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`bool std::is_permutation (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2, __↵`
`ForwardIterator2 __last2, _BinaryPredicate __pred)`
- `template<typename _ForwardIterator >`
`bool std::is_sorted (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`bool std::is_sorted (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _ForwardIterator >`
`_ForwardIterator std::is_sorted_until (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`_ForwardIterator std::is_sorted_until (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`_ForwardIterator std::lower_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __val, __↵`
`Compare __comp)`
- `template<typename _Tp >`
`_GLIBCXX14_CONSTEXPR _Tp std::max (initializer_list< _Tp >)`
- `template<typename _Tp, typename _Compare >`
`_GLIBCXX14_CONSTEXPR _Tp std::max (initializer_list< _Tp >, _Compare)`
- `template<typename _ForwardIterator >`
`_GLIBCXX14_CONSTEXPR _ForwardIterator std::max_element (_ForwardIterator __first, _ForwardIterator __↵`
`last)`
- `template<typename _ForwardIterator, typename _Compare >`
`_GLIBCXX14_CONSTEXPR _ForwardIterator std::max_element (_ForwardIterator __first, _ForwardIterator __↵`
`last, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`_OutputIterator std::merge (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Input↵`
`Iterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::merge (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Input↵`
`Iterator2 __last2, _OutputIterator __result, _Compare __comp)`

- `template<typename _Tp >`
`_GLIBCXX14_CONSTEXPR _Tp std::min (initializer_list< _Tp >)`
- `template<typename _Tp, typename _Compare >`
`_GLIBCXX14_CONSTEXPR _Tp std::min (initializer_list< _Tp >, _Compare)`
- `template<typename _ForwardIterator >`
`_GLIBCXX14_CONSTEXPR _ForwardIterator std::min_element (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`_GLIBCXX14_CONSTEXPR _ForwardIterator std::min_element (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _Tp >`
`_GLIBCXX14_CONSTEXPR pair< const _Tp &, const _Tp & > std::minmax (const _Tp &__a, const _Tp &__b)`
- `template<typename _Tp, typename _Compare >`
`_GLIBCXX14_CONSTEXPR pair< const _Tp &, const _Tp & > std::minmax (const _Tp &__a, const _Tp &__b, _Compare __comp)`
- `template<typename _Tp >`
`_GLIBCXX14_CONSTEXPR pair< _Tp, _Tp > std::minmax (initializer_list< _Tp >)`
- `template<typename _Tp, typename _Compare >`
`_GLIBCXX14_CONSTEXPR pair< _Tp, _Tp > std::minmax (initializer_list< _Tp >, _Compare)`
- `template<typename _ForwardIterator >`
`_GLIBCXX14_CONSTEXPR pair< _ForwardIterator, _ForwardIterator > std::minmax_element (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Compare >`
`_GLIBCXX14_CONSTEXPR pair< _ForwardIterator, _ForwardIterator > std::minmax_element (_ForwardIterator __first, _ForwardIterator __last, _Compare __comp)`
- `template<typename _BidirectionalIterator >`
`bool std::next_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`bool std::next_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare __comp)`
- `template<typename _InputIterator, typename _Predicate >`
`bool std::none_of (_InputIterator __first, _InputIterator __last, _Predicate __pred)`
- `template<typename _RandomAccessIterator >`
`void std::nth_element (_RandomAccessIterator __first, _RandomAccessIterator __nth, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::nth_element (_RandomAccessIterator __first, _RandomAccessIterator __nth, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`void std::partial_sort (_RandomAccessIterator __first, _RandomAccessIterator __middle, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::partial_sort (_RandomAccessIterator __first, _RandomAccessIterator __middle, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _InputIterator, typename _RandomAccessIterator >`
`_RandomAccessIterator std::partial_sort_copy (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __result_first, _RandomAccessIterator __result_last)`
- `template<typename _InputIterator, typename _RandomAccessIterator, typename _Compare >`
`_RandomAccessIterator std::partial_sort_copy (_InputIterator __first, _InputIterator __last, _RandomAccessIterator __result_first, _RandomAccessIterator __result_last, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator std::partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`

- `template<typename _InputIterator, typename _OutputIterator1, typename _OutputIterator2, typename _Predicate >`
`pair< _OutputIterator1, _OutputIterator2 > std::partition_copy (_InputIterator __first, _InputIterator __last, ↵`
`_OutputIterator1 __out_true, _OutputIterator2 __out_false, _Predicate __pred)`
- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator std::partition_point (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _BidirectionalIterator >`
`bool std::prev_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last)`
- `template<typename _BidirectionalIterator, typename _Compare >`
`bool std::prev_permutation (_BidirectionalIterator __first, _BidirectionalIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _RandomNumberGenerator >`
`void std::random_shuffle (_RandomAccessIterator __first, _RandomAccessIterator __last, _RandomNumber↵`
`Generator &&__rand)`
- `template<typename _ForwardIterator, typename _Tp >`
`_ForwardIterator std::remove (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Tp >`
`_OutputIterator std::remove_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result, const _Tp ↵`
`&__value)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate >`
`_OutputIterator std::remove_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, ↵`
`_Predicate __pred)`
- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator std::remove_if (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _ForwardIterator, typename _Tp >`
`void std::replace (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__old_value, const _Tp &__new_↵`
`__value)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Tp >`
`_OutputIterator std::replace_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result, const _Tp ↵`
`&__old_value, const _Tp &__new_value)`
- `template<typename _InputIterator, typename _OutputIterator, typename _Predicate, typename _Tp >`
`_OutputIterator std::replace_copy_if (_InputIterator __first, _InputIterator __last, _OutputIterator __result, ↵`
`_Predicate __pred, const _Tp &__new_value)`
- `template<typename _ForwardIterator, typename _Predicate, typename _Tp >`
`void std::replace_if (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred, const _Tp &__new_↵`
`__value)`
- `template<typename _BidirectionalIterator >`
`void std::reverse (_BidirectionalIterator __first, _BidirectionalIterator __last)`
- `template<typename _BidirectionalIterator, typename _OutputIterator >`
`_OutputIterator std::reverse_copy (_BidirectionalIterator __first, _BidirectionalIterator __last, _OutputIterator ↵`
`__result)`
- `template<typename _ForwardIterator >`
`_ForwardIterator std::_V2::rotate (_ForwardIterator __first, _ForwardIterator __middle, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _OutputIterator >`
`_OutputIterator std::rotate_copy (_ForwardIterator __first, _ForwardIterator __middle, _ForwardIterator __last, ↵`
`_OutputIterator __result)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`_ForwardIterator1 std::search (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2,`
`_ForwardIterator2 __last2)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2, typename _BinaryPredicate >`
`_ForwardIterator1 std::search (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2,`
`_ForwardIterator2 __last2, _BinaryPredicate __predicate)`
- `template<typename _ForwardIterator, typename _Integer, typename _Tp >`
`_ForwardIterator std::search_n (_ForwardIterator __first, _ForwardIterator __last, _Integer __count, const _Tp ↵`
`&__val)`

- `template<typename _ForwardIterator, typename _Integer, typename _Tp, typename _BinaryPredicate >`
`_ForwardIterator std::search_n (_ForwardIterator __first, _ForwardIterator __last, _Integer __count, const _Tp &__val, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`_OutputIterator std::set_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, ↵
_InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::set_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, ↵
_InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`_OutputIterator std::set_intersection (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, ↵
_InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::set_intersection (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, ↵
_InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`_OutputIterator std::set_symmetric_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 ↵
__first2, _InputIterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::set_symmetric_difference (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 ↵
__first2, _InputIterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator >`
`_OutputIterator std::set_union (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Input↵
Iterator2 __last2, _OutputIterator __result)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _Compare >`
`_OutputIterator std::set_union (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Input↵
Iterator2 __last2, _OutputIterator __result, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _UniformRandomNumberGenerator >`
`void std::shuffle (_RandomAccessIterator __first, _RandomAccessIterator __last, _UniformRandomNumber↵
Generator &&__g)`
- `template<typename _RandomAccessIterator >`
`void std::sort (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Predicate >`
`_ForwardIterator std::stable_partition (_ForwardIterator __first, _ForwardIterator __last, _Predicate __pred)`
- `template<typename _RandomAccessIterator >`
`void std::stable_sort (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::stable_sort (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _InputIterator, typename _OutputIterator, typename _UnaryOperation >`
`_OutputIterator std::transform (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _Unary↵
Operation __unary_op)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::transform (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _Output↵
Iterator __result, _BinaryOperation __binary_op)`
- `template<typename _ForwardIterator >`
`_ForwardIterator std::unique (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _BinaryPredicate >`
`_ForwardIterator std::unique (_ForwardIterator __first, _ForwardIterator __last, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator, typename _OutputIterator >`
`_OutputIterator std::unique_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result)`

- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryPredicate >`
`_OutputIterator std::unique_copy (_InputIterator __first, _InputIterator __last, _OutputIterator __result, _BinaryPredicate __binary_pred)`
- `template<typename _ForwardIterator, typename _Tp >`
`_ForwardIterator std::upper_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`_ForwardIterator std::upper_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare __comp)`

6.569.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<algorithm>`.

6.569.2 Function Documentation

6.569.2.1 `template<typename _ForwardIterator > _ForwardIterator std::_V2::__rotate (_ForwardIterator __first, _ForwardIterator __middle, _ForwardIterator __last, forward_iterator_tag)`

This is a helper function for the rotate algorithm.

Definition at line 1246 of file `stl_algo.h`.

References `std::_V2::__rotate()`, and `std::iter_swap()`.

Referenced by `std::_V2::__rotate()`.

6.569.2.2 `template<typename _BidirectionalIterator > _BidirectionalIterator std::_V2::__rotate (_BidirectionalIterator __first, _BidirectionalIterator __middle, _BidirectionalIterator __last, bidirectional_iterator_tag)`

This is a helper function for the rotate algorithm.

Definition at line 1287 of file `stl_algo.h`.

References `std::__reverse()`, `std::_V2::__rotate()`, and `std::iter_swap()`.

6.569.2.3 `template<typename _RandomAccessIterator > _RandomAccessIterator std::_V2::__rotate (_RandomAccessIterator __first, _RandomAccessIterator __middle, _RandomAccessIterator __last, random_access_iterator_tag)`

This is a helper function for the rotate algorithm.

Definition at line 1325 of file `stl_algo.h`.

References `std::_V2::__rotate()`, `std::iter_swap()`, and `std::swap_ranges()`.

6.570 `stl_algobase.h` File Reference

Classes

- struct `std::char_traits<_CharT>`
- class `std::istreambuf_iterator<_CharT, _Traits>`
- class `std::ostreambuf_iterator<_CharT, _Traits>`

Namespaces

- `std`

Macros

- `#define __cpp_lib_robust_nonmodifying_seq_ops`
- `#define _GLIBCXX_MOVE3(_Tp, _Up, _Vp)`
- `#define _GLIBCXX_MOVE_BACKWARD3(_Tp, _Up, _Vp)`

Functions

- `template<bool _IsMove, typename _II, typename _OI>`
`_OI std::__copy_move_a (_II __first, _II __last, _OI __result)`
- `template<bool _IsMove, typename _CharT>`
`__gnu_cxx::__enable_if<__is_char<_CharT>::__value, ostreambuf_iterator<_CharT, char_traits<_CharT>>`
`> >::__type std::__copy_move_a2 (_CharT *, _CharT *, ostreambuf_iterator<_CharT, char_traits<_CharT>`
`> >)`
- `template<bool _IsMove, typename _CharT>`
`__gnu_cxx::__enable_if<__is_char<_CharT>::__value, ostreambuf_iterator<_CharT, char_traits<_CharT>`
`> > >::__type std::__copy_move_a2 (const _CharT *, const _CharT *, ostreambuf_iterator<_CharT, char_`
`traits<_CharT> > >)`
- `template<bool _IsMove, typename _CharT>`
`__gnu_cxx::__enable_if<__is_char<_CharT>::__value, _CharT * >::__type std::__copy_move_a2`
`(istreambuf_iterator<_CharT, char_traits<_CharT> > >, istreambuf_iterator<_CharT, char_traits<_CharT>`
`> >, _CharT *)`
- `template<bool _IsMove, typename _II, typename _OI>`
`_OI std::__copy_move_a2 (_II __first, _II __last, _OI __result)`
- `template<bool _IsMove, typename _BI1, typename _BI2>`
`_BI2 std::__copy_move_backward_a (_BI1 __first, _BI1 __last, _BI2 __result)`
- `template<bool _IsMove, typename _BI1, typename _BI2>`
`_BI2 std::__copy_move_backward_a2 (_BI1 __first, _BI1 __last, _BI2 __result)`
- `template<typename _II1, typename _II2>`
`bool std::__equal_aux (_II1 __first1, _II1 __last1, _II2 __first2)`
- `template<typename _ForwardIterator, typename _Tp>`
`__gnu_cxx::__enable_if<!__is_scalar<_Tp>::__value, void >::__type std::__fill_a (_ForwardIterator __first,`
`_ForwardIterator __last, const _Tp &__value)`
- `template<typename _ForwardIterator, typename _Tp>`
`__gnu_cxx::__enable_if<__is_scalar<_Tp>::__value, void >::__type std::__fill_a (_ForwardIterator __first,`
`_ForwardIterator __last, const _Tp &__value)`

- `template<typename _Tp >`
`__gnu_cxx::__enable_if< __is_byte< _Tp >::__value, void >::__type std::__fill_a (_Tp *__first, _Tp *__last,`
`const _Tp &__c)`
- `template<typename _OutputIterator, typename _Size, typename _Tp >`
`__gnu_cxx::__enable_if<!__is_scalar< _Tp >::__value, _OutputIterator >::__type std::__fill_n_a (_OutputIterator`
`__first, _Size __n, const _Tp &__value)`
- `template<typename _OutputIterator, typename _Size, typename _Tp >`
`__gnu_cxx::__enable_if< __is_scalar< _Tp >::__value, _OutputIterator >::__type std::__fill_n_a (_OutputIterator`
`__first, _Size __n, const _Tp &__value)`
- `template<typename _Size, typename _Tp >`
`__gnu_cxx::__enable_if< __is_byte< _Tp >::__value, _Tp * >::__type std::__fill_n_a (_Tp *__first, _Size __n,`
`const _Tp &__c)`
- `template<typename _II1, typename _II2 >`
`bool std::__lexicographical_compare_aux (_II1 __first1, _II1 __last1, _II2 __first2, _II2 __last2)`
- `template<typename _II1, typename _II2, typename _Compare >`
`bool std::__lexicographical_compare_impl (_II1 __first1, _II1 __last1, _II2 __first2, _II2 __last2, _Compare`
`__comp)`
- `constexpr int std::__lg (int __n)`
- `constexpr unsigned std::__lg (unsigned __n)`
- `constexpr long std::__lg (long __n)`
- `constexpr unsigned long std::__lg (unsigned long __n)`
- `constexpr long long std::__lg (long long __n)`
- `constexpr unsigned long long std::__lg (unsigned long long __n)`
- `template<typename _ForwardIterator, typename _Tp, typename _Compare >`
`_ForwardIterator std::__lower_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__val, _Compare`
`__comp)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`pair< _InputIterator1, _InputIterator2 > std::__mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2`
`__first2, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`pair< _InputIterator1, _InputIterator2 > std::__mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2`
`__first2, _InputIterator2 __last2, _BinaryPredicate __binary_pred)`
- `template<typename _Iterator >`
`_Iterator std::__niter_base (_Iterator __it)`
- `template<typename _II, typename _OI >`
`_OI std::__copy (_II __first, _II __last, _OI __result)`
- `template<typename _BI1, typename _BI2 >`
`_BI2 std::__copy_backward (_BI1 __first, _BI1 __last, _BI2 __result)`
- `template<typename _IIter1, typename _IIter2, typename _BinaryPredicate >`
`bool std::__equal (_IIter1 __first1, _IIter1 __last1, _IIter2 __first2, _BinaryPredicate __binary_pred)`
- `template<typename _II1, typename _II2 >`
`bool std::__equal (_II1 __first1, _II1 __last1, _II2 __first2)`
- `template<typename _II1, typename _II2 >`
`bool std::__equal (_II1 __first1, _II1 __last1, _II2 __first2, _II2 __last2)`
- `template<typename _IIter1, typename _IIter2, typename _BinaryPredicate >`
`bool std::__equal (_IIter1 __first1, _IIter1 __last1, _IIter2 __first2, _IIter2 __last2, _BinaryPredicate __binary_pred)`
- `template<typename _ForwardIterator, typename _Tp >`
`void std::fill (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__value)`
- `template<typename _OI, typename _Size, typename _Tp >`
`_OI std::fill_n (_OI __first, _Size __n, const _Tp &__value)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`void std::iter_swap (_ForwardIterator1 __a, _ForwardIterator2 __b)`

- `template<typename _II1, typename _II2 >`
`bool std::lexicographical_compare (_II1 __first1, _II1 __last1, _II2 __first2, _II2 __last2)`
- `template<typename _II1, typename _II2, typename _Compare >`
`bool std::lexicographical_compare (_II1 __first1, _II1 __last1, _II2 __first2, _II2 __last2, _Compare __comp)`
- `template<typename _ForwardIterator, typename _Tp >`
`_ForwardIterator std::lower_bound (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __val)`
- `template<typename _Tp >`
`_GLIBCXX14_CONSTEXPR const _Tp & std::max (const _Tp & __a, const _Tp & __b)`
- `template<typename _Tp, typename _Compare >`
`_GLIBCXX14_CONSTEXPR const _Tp & std::max (const _Tp & __a, const _Tp & __b, _Compare __comp)`
- `template<typename _Tp >`
`_GLIBCXX14_CONSTEXPR const _Tp & std::min (const _Tp & __a, const _Tp & __b)`
- `template<typename _Tp, typename _Compare >`
`_GLIBCXX14_CONSTEXPR const _Tp & std::min (const _Tp & __a, const _Tp & __b, _Compare __comp)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`pair< _InputIterator1, _InputIterator2 > std::mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`pair< _InputIterator1, _InputIterator2 > std::mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _BinaryPredicate __binary_pred)`
- `template<typename _InputIterator1, typename _InputIterator2 >`
`pair< _InputIterator1, _InputIterator2 > std::mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _BinaryPredicate >`
`pair< _InputIterator1, _InputIterator2 > std::mismatch (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _BinaryPredicate __binary_pred)`
- `template<typename _II, typename _OI >`
`_OI std::move (_II __first, _II __last, _OI __result)`
- `template<typename _BI1, typename _BI2 >`
`_BI2 std::move_backward (_BI1 __first, _BI1 __last, _BI2 __result)`
- `template<typename _ForwardIterator1, typename _ForwardIterator2 >`
`_ForwardIterator2 std::swap_ranges (_ForwardIterator1 __first1, _ForwardIterator1 __last1, _ForwardIterator2 __first2)`

6.570.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<algorithm>`.

6.571 `std_bvector.h` File Reference

Classes

- struct [std::hash<::vector< bool, _Alloc > >](#)
- class [std::vector< bool, _Alloc >](#)

Namespaces

- [std](#)

Typedefs

- typedef unsigned long **std::_Bit_type**

Enumerations

- enum { **_S_word_bit** }

Functions

- void **std::__fill_bvector** (_Bit_iterator __first, _Bit_iterator __last, bool __x)
- void **std::fill** (_Bit_iterator __first, _Bit_iterator __last, const bool &__x)
- _Bit_iterator **std::operator+** (ptrdiff_t __n, const _Bit_iterator &__x)
- _Bit_const_iterator **std::operator+** (ptrdiff_t __n, const _Bit_const_iterator &__x)
- ptrdiff_t **std::operator-** (const _Bit_iterator_base &__x, const _Bit_iterator_base &__y)
- void **std::swap** (_Bit_reference __x, _Bit_reference __y) noexcept
- void **std::swap** (_Bit_reference __x, bool &__y) noexcept
- void **std::swap** (bool &__x, _Bit_reference __y) noexcept

6.571.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<vector>`.

6.572 stl_construct.h File Reference

Namespaces

- [std](#)

Functions

- template<typename _T1, typename... _Args>
void [std::_Construct](#) (_T1 *__p, _Args &&... __args)
- template<typename _Tp>
void [std::_Destroy](#) (_Tp *__pointer)
- template<typename _ForwardIterator>
void [std::_Destroy](#) (_ForwardIterator __first, _ForwardIterator __last)
- template<typename _ForwardIterator, typename _Allocator>
void [std::_Destroy](#) (_ForwardIterator __first, _ForwardIterator __last, _Allocator &__alloc)
- template<typename _ForwardIterator, typename _Tp>
void **std::_Destroy** (_ForwardIterator __first, _ForwardIterator __last, allocator<_Tp> &)

6.572.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

6.573 `std_deque.h` File Reference

Classes

- class `std::_Deque_base<_Tp, _Alloc>`
- struct `std::_Deque_iterator<_Tp, _Ref, _Ptr>`
- class `std::deque<_Tp, _Alloc>`

Namespaces

- `std`

Macros

- `#define _GLIBCXX_DEQUE_BUF_SIZE`

Functions

- constexpr `size_t std::__deque_buf_size (size_t __size)`
- template<typename `_Tp`>
`_Deque_iterator<_Tp, _Tp &, _Tp * > std::copy (_Deque_iterator<_Tp, _Tp &, _Tp * > __first, _Deque_iterator<_Tp, _Tp &, _Tp * > __last, _Deque_iterator<_Tp, _Tp &, _Tp * > __result)`
- template<typename `_Tp`>
`_Deque_iterator<_Tp, _Tp &, _Tp * > std::copy (_Deque_iterator<_Tp, const _Tp &, const _Tp * > __first, _Deque_iterator<_Tp, const _Tp &, const _Tp * > __last, _Deque_iterator<_Tp, _Tp &, _Tp * > __result)`
- template<typename `_Tp`>
`_Deque_iterator<_Tp, _Tp &, _Tp * > std::copy_backward (_Deque_iterator<_Tp, _Tp &, _Tp * > __first, _Deque_iterator<_Tp, _Tp &, _Tp * > __last, _Deque_iterator<_Tp, _Tp &, _Tp * > __result)`
- template<typename `_Tp`>
`_Deque_iterator<_Tp, _Tp &, _Tp * > std::copy_backward (_Deque_iterator<_Tp, const _Tp &, const _Tp * > __first, _Deque_iterator<_Tp, const _Tp &, const _Tp * > __last, _Deque_iterator<_Tp, _Tp &, _Tp * > __result)`
- template<typename `_Tp`>
`void std::fill (const _Deque_iterator<_Tp, _Tp &, _Tp * > &__first, const _Deque_iterator<_Tp, _Tp &, _Tp * > &__last, const _Tp &__value)`
- template<typename `_Tp`>
`_Deque_iterator<_Tp, _Tp &, _Tp * > std::move (_Deque_iterator<_Tp, _Tp &, _Tp * > __first, _Deque_iterator<_Tp, _Tp &, _Tp * > __last, _Deque_iterator<_Tp, _Tp &, _Tp * > __result)`
- template<typename `_Tp`>
`_Deque_iterator<_Tp, _Tp &, _Tp * > std::move (_Deque_iterator<_Tp, const _Tp &, const _Tp * > __first, _Deque_iterator<_Tp, const _Tp &, const _Tp * > __last, _Deque_iterator<_Tp, _Tp &, _Tp * > __result)`

- `template<typename _Tp >`
`_Deque_iterator< _Tp, _Tp &, _Tp * > std::move_backward (_Deque_iterator< _Tp, _Tp &, _Tp * > __first,`
`_Deque_iterator< _Tp, _Tp &, _Tp * > __last, _Deque_iterator< _Tp, _Tp &, _Tp * > __result)`
- `template<typename _Tp >`
`_Deque_iterator< _Tp, _Tp &, _Tp * > std::move_backward (_Deque_iterator< _Tp, const _Tp &, const _Tp`
`* > __first, _Deque_iterator< _Tp, const _Tp &, const _Tp * > __last, _Deque_iterator< _Tp, _Tp &, _Tp * >`
`__result)`
- `template<typename _Tp, typename _Ref, typename _Ptr >`
`bool std::operator!= (const _Deque_iterator< _Tp, _Ref, _Ptr > &__x, const _Deque_iterator< _Tp, _Ref, _Ptr`
`> &__y) noexcept`
- `template<typename _Tp, typename _RefL, typename _PtrL, typename _RefR, typename _PtrR >`
`bool std::operator!= (const _Deque_iterator< _Tp, _RefL, _PtrL > &__x, const _Deque_iterator< _Tp, _RefR,`
`_PtrR > &__y) noexcept`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator!= (const deque< _Tp, _Alloc > &__x, const deque< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Ref, typename _Ptr >`
`_Deque_iterator< _Tp, _Ref, _Ptr > std::operator+ (ptrdiff_t __n, const _Deque_iterator< _Tp, _Ref, _Ptr >`
`&__x) noexcept`
- `template<typename _Tp, typename _Ref, typename _Ptr >`
`_Deque_iterator< _Tp, _Ref, _Ptr >::difference_type std::operator- (const _Deque_iterator< _Tp, _Ref, _Ptr >`
`&__x, const _Deque_iterator< _Tp, _Ref, _Ptr > &__y) noexcept`
- `template<typename _Tp, typename _RefL, typename _PtrL, typename _RefR, typename _PtrR >`
`_Deque_iterator< _Tp, _RefL, _PtrL >::difference_type std::operator- (const _Deque_iterator< _Tp, _RefL,`
`_PtrL > &__x, const _Deque_iterator< _Tp, _RefR, _PtrR > &__y) noexcept`
- `template<typename _Tp, typename _Ref, typename _Ptr >`
`bool std::operator< (const _Deque_iterator< _Tp, _Ref, _Ptr > &__x, const _Deque_iterator< _Tp, _Ref, _Ptr`
`> &__y) noexcept`
- `template<typename _Tp, typename _RefL, typename _PtrL, typename _RefR, typename _PtrR >`
`bool std::operator< (const _Deque_iterator< _Tp, _RefL, _PtrL > &__x, const _Deque_iterator< _Tp, _RefR,`
`_PtrR > &__y) noexcept`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator< (const deque< _Tp, _Alloc > &__x, const deque< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Ref, typename _Ptr >`
`bool std::operator<= (const _Deque_iterator< _Tp, _Ref, _Ptr > &__x, const _Deque_iterator< _Tp, _Ref, _Ptr`
`> &__y) noexcept`
- `template<typename _Tp, typename _RefL, typename _PtrL, typename _RefR, typename _PtrR >`
`bool std::operator<= (const _Deque_iterator< _Tp, _RefL, _PtrL > &__x, const _Deque_iterator< _Tp, _RefR,`
`_PtrR > &__y) noexcept`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator<= (const deque< _Tp, _Alloc > &__x, const deque< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Ref, typename _Ptr >`
`bool std::operator== (const _Deque_iterator< _Tp, _Ref, _Ptr > &__x, const _Deque_iterator< _Tp, _Ref, _Ptr`
`> &__y) noexcept`
- `template<typename _Tp, typename _RefL, typename _PtrL, typename _RefR, typename _PtrR >`
`bool std::operator== (const _Deque_iterator< _Tp, _RefL, _PtrL > &__x, const _Deque_iterator< _Tp, _RefR,`
`_PtrR > &__y) noexcept`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator== (const deque< _Tp, _Alloc > &__x, const deque< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Ref, typename _Ptr >`
`bool std::operator> (const _Deque_iterator< _Tp, _Ref, _Ptr > &__x, const _Deque_iterator< _Tp, _Ref, _Ptr`
`> &__y) noexcept`
- `template<typename _Tp, typename _RefL, typename _PtrL, typename _RefR, typename _PtrR >`
`bool std::operator> (const _Deque_iterator< _Tp, _RefL, _PtrL > &__x, const _Deque_iterator< _Tp, _RefR,`
`_PtrR > &__y) noexcept`

- `template<typename _Tp, typename _Alloc >`
`bool std::operator< (const deque< _Tp, _Alloc > &__x, const deque< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Ref, typename _Ptr >`
`bool std::operator>= (const _Deque_iterator< _Tp, _Ref, _Ptr > &__x, const _Deque_iterator< _Tp, _Ref, _Ptr > &__y) noexcept`
- `template<typename _Tp, typename _RefL, typename _PtrL, typename _RefR, typename _PtrR >`
`bool std::operator>= (const _Deque_iterator< _Tp, _RefL, _PtrL > &__x, const _Deque_iterator< _Tp, _RefR, _PtrR > &__y) noexcept`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator>= (const deque< _Tp, _Alloc > &__x, const deque< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`void std::swap (deque< _Tp, _Alloc > &__x, deque< _Tp, _Alloc > &__y) noexcept(/*conditional */)`

6.573.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<deque>`.

6.573.2 Macro Definition Documentation

6.573.2.1 `#define _GLIBCXX_DEQUE_BUF_SIZE`

This function controls the size of memory nodes.

Parameters

<code>__size</code>	The size of an element.
---------------------	-------------------------

Returns

The number (not byte size) of elements per node.

This function started off as a compiler kludge from SGI, but seems to be a useful wrapper around a repeated constant expression. The **512** is tunable (and no other code needs to change), but no investigation has been done since inheriting the SGI code. Touch `_GLIBCXX_DEQUE_BUF_SIZE` only if you know what you are doing, however: changing it breaks the binary compatibility!!

Definition at line 85 of file `std_deque.h`.

6.574 `std_function.h` File Reference

Classes

- struct [std::binary_function< _Arg1, _Arg2, _Result >](#)
- class [std::binary_negate< _Predicate >](#)

- class `std::const_mem_fun1_ref_t<_Ret, _Tp, _Arg>`
- class `std::const_mem_fun1_t<_Ret, _Tp, _Arg>`
- class `std::const_mem_fun_ref_t<_Ret, _Tp>`
- class `std::const_mem_fun_t<_Ret, _Tp>`
- struct `std::divides<_Tp>`
- struct `std::divides<_Tp>`
- struct `std::divides<void>`
- struct `std::equal_to<_Tp>`
- struct `std::equal_to<_Tp>`
- struct `std::equal_to<void>`
- struct `std::greater<_Tp>`
- struct `std::greater<_Tp>`
- struct `std::greater<void>`
- struct `std::greater_equal<_Tp>`
- struct `std::greater_equal<_Tp>`
- struct `std::greater_equal<void>`
- struct `std::less<_Tp>`
- struct `std::less<_Tp>`
- struct `std::less<void>`
- struct `std::less_equal<_Tp>`
- struct `std::less_equal<_Tp>`
- struct `std::less_equal<void>`
- struct `std::logical_and<_Tp>`
- struct `std::logical_and<_Tp>`
- struct `std::logical_and<void>`
- struct `std::logical_not<_Tp>`
- struct `std::logical_not<_Tp>`
- struct `std::logical_not<void>`
- struct `std::logical_or<_Tp>`
- struct `std::logical_or<_Tp>`
- struct `std::logical_or<void>`
- class `std::mem_fun1_ref_t<_Ret, _Tp, _Arg>`
- class `std::mem_fun1_t<_Ret, _Tp, _Arg>`
- class `std::mem_fun_ref_t<_Ret, _Tp>`
- class `std::mem_fun_t<_Ret, _Tp>`
- struct `std::minus<_Tp>`
- struct `std::minus<_Tp>`
- struct `std::minus<void>`
- struct `std::modulus<_Tp>`
- struct `std::modulus<_Tp>`
- struct `std::modulus<void>`
- struct `std::multiplies<_Tp>`
- struct `std::multiplies<_Tp>`
- struct `std::multiplies<void>`
- struct `std::negate<_Tp>`
- struct `std::negate<_Tp>`
- struct `std::negate<void>`
- struct `std::not_equal_to<_Tp>`
- struct `std::not_equal_to<_Tp>`
- struct `std::not_equal_to<void>`
- struct `std::plus<_Tp>`

- struct `std::plus<_Tp>`
- class `std::pointer_to_binary_function<_Arg1, _Arg2, _Result>`
- class `std::pointer_to_unary_function<_Arg, _Result>`
- struct `std::unary_function<_Arg, _Result>`
- class `std::unary_negate<_Predicate>`

Namespaces

- `std`

Macros

- `#define __cpp_lib_transparent_operators`

Functions

- `template<typename _Ret, typename _Tp>`
`mem_fun_t<_Ret, _Tp> std::mem_fun (_Ret(_Tp::*__f)())`
- `template<typename _Ret, typename _Tp>`
`const_mem_fun_t<_Ret, _Tp> std::mem_fun (_Ret(_Tp::*__f)() const)`
- `template<typename _Ret, typename _Tp, typename _Arg>`
`mem_fun1_t<_Ret, _Tp, _Arg> std::mem_fun (_Ret(_Tp::*__f)(_Arg))`
- `template<typename _Ret, typename _Tp, typename _Arg>`
`const_mem_fun1_t<_Ret, _Tp, _Arg> std::mem_fun (_Ret(_Tp::*__f)(_Arg) const)`
- `template<typename _Ret, typename _Tp>`
`mem_fun_ref_t<_Ret, _Tp> std::mem_fun_ref (_Ret(_Tp::*__f)())`
- `template<typename _Ret, typename _Tp>`
`const_mem_fun_ref_t<_Ret, _Tp> std::mem_fun_ref (_Ret(_Tp::*__f)() const)`
- `template<typename _Ret, typename _Tp, typename _Arg>`
`mem_fun1_ref_t<_Ret, _Tp, _Arg> std::mem_fun_ref (_Ret(_Tp::*__f)(_Arg))`
- `template<typename _Ret, typename _Tp, typename _Arg>`
`const_mem_fun1_ref_t<_Ret, _Tp, _Arg> std::mem_fun_ref (_Ret(_Tp::*__f)(_Arg) const)`
- `template<typename _Predicate>`
`_GLIBCXX14_CONSTEXPR unary_negate<_Predicate> std::not1 (const _Predicate &__pred)`
- `template<typename _Predicate>`
`_GLIBCXX14_CONSTEXPR binary_negate<_Predicate> std::not2 (const _Predicate &__pred)`
- `template<typename _Arg, typename _Result>`
`pointer_to_unary_function<_Arg, _Result> std::ptr_fun (_Result(*__x)(_Arg))`
- `template<typename _Arg1, typename _Arg2, typename _Result>`
`pointer_to_binary_function<_Arg1, _Arg2, _Result> std::ptr_fun (_Result(*__x)(_Arg1, _Arg2))`

6.574.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<functional>`.

6.575 std_heap.h File Reference

Namespaces

- [std](#)

Functions

- `template<typename _RandomAccessIterator, typename _Distance, typename _Tp, typename _Compare >`
`void std::adjust_heap (_RandomAccessIterator __first, _Distance __holeIndex, _Distance __len, _Tp __↵`
`value, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Distance >`
`bool std::is_heap (_RandomAccessIterator __first, _Distance __n)`
- `template<typename _RandomAccessIterator, typename _Compare, typename _Distance >`
`bool std::is_heap (_RandomAccessIterator __first, _Compare __comp, _Distance __n)`
- `template<typename _RandomAccessIterator >`
`bool std::is_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`bool std::is_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Distance, typename _Compare >`
`_Distance std::is_heap_until (_RandomAccessIterator __first, _Distance __n, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::make_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::pop_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _RandomAccessIterator`
`__result, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Distance, typename _Tp, typename _Compare >`
`void std::push_heap (_RandomAccessIterator __first, _Distance __holeIndex, _Distance __topIndex, _Tp __↵`
`__value, _Compare __comp)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::sort_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`bool std::is_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`bool std::is_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`_RandomAccessIterator std::is_heap_until (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`_RandomAccessIterator std::is_heap_until (_RandomAccessIterator __first, _RandomAccessIterator __last, _↵`
`Compare __comp)`
- `template<typename _RandomAccessIterator >`
`void std::make_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::make_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`void std::pop_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::pop_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`void std::push_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`

- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::push_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`
- `template<typename _RandomAccessIterator >`
`void std::sort_heap (_RandomAccessIterator __first, _RandomAccessIterator __last)`
- `template<typename _RandomAccessIterator, typename _Compare >`
`void std::sort_heap (_RandomAccessIterator __first, _RandomAccessIterator __last, _Compare __comp)`

6.575.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<queue>`.

6.576 `std_iterator.h` File Reference

Classes

- class [std::back_insert_iterator](#)< _Container >
- class [std::front_insert_iterator](#)< _Container >
- class [std::insert_iterator](#)< _Container >
- class [std::move_iterator](#)< _Iterator >
- class [std::reverse_iterator](#)< _Iterator >

Namespaces

- [__gnu_cxx](#)
- [std](#)

Macros

- `#define __cpp_lib_make_reverse_iterator`
- `#define _GLIBCXX_MAKE_MOVE_IF_NOEXCEPT_ITERATOR(_Iter)`
- `#define _GLIBCXX_MAKE_MOVE_ITERATOR(_Iter)`

Functions

- `template<typename _Iterator, typename _ReturnType = typename conditional<__move_if_noexcept_cond<typename iterator_traits<_Iterator>::value_type>::value, _Iterator, move_iterator<_Iterator>>::type>`
`_ReturnType std::__make_move_if_noexcept_iterator (_Iterator __i)`
- `template<typename _Tp, typename _ReturnType = typename conditional<__move_if_noexcept_cond<_Tp>::value, const _Tp*, move_iterator<_Tp*>>::type>`
`_ReturnType std::__make_move_if_noexcept_iterator (_Tp * __i)`
- `template<typename _Iterator >`
`reverse_iterator<_Iterator > std::__make_reverse_iterator (_Iterator __i)`
- `template<typename _Iterator >`
`auto std::__miter_base (reverse_iterator<_Iterator > __it) -> decltype(__make_reverse_iterator(__miter_base(__it.base())))`

- `template<typename _Iterator >`
`auto std::__miter_base (move_iterator< _Iterator > __it) -> decltype(__miter_base(__it.base()))`
- `template<typename _Iterator >`
`auto std::__niter_base (reverse_iterator< _Iterator > __it) -> decltype(__make_reverse_iterator(__niter_base(__it.base())))`
- `template<typename _Iterator, typename _Container >`
`_Iterator std::__niter_base (__gnu_cxx::__normal_iterator< _Iterator, _Container > __it)`
- `template<typename _Iterator >`
`auto std::__niter_base (move_iterator< _Iterator > __it) -> decltype(make_move_iterator(__niter_base(__it.base())))`
- `template<typename _Container >`
`back_insert_iterator< _Container > std::back_inserter (_Container & __x)`
- `template<typename _Container >`
`front_insert_iterator< _Container > std::front_inserter (_Container & __x)`
- `template<typename _Container, typename _Iterator >`
`insert_iterator< _Container > std::inserter (_Container & __x, _Iterator __i)`
- `template<typename _Iterator >`
`move_iterator< _Iterator > std::make_move_iterator (_Iterator __i)`
- `template<typename _Iterator >`
`reverse_iterator< _Iterator > std::make_reverse_iterator (_Iterator __i)`
- `template<typename _Iterator >`
`bool std::operator!= (const reverse_iterator< _Iterator > & __x, const reverse_iterator< _Iterator > & __y)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool std::operator!= (const reverse_iterator< _IteratorL > & __x, const reverse_iterator< _IteratorR > & __y)`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`bool __gnu_cxx::operator!= (const __normal_iterator< _IteratorL, _Container > & __lhs, const __normal_iterator< _IteratorR, _Container > & __rhs) noexcept`
- `template<typename _Iterator, typename _Container >`
`bool __gnu_cxx::operator!= (const __normal_iterator< _Iterator, _Container > & __lhs, const __normal_iterator< _Iterator, _Container > & __rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR >`
`bool std::operator!= (const move_iterator< _IteratorL > & __x, const move_iterator< _IteratorR > & __y)`
- `template<typename _Iterator >`
`bool std::operator!= (const move_iterator< _Iterator > & __x, const move_iterator< _Iterator > & __y)`
- `template<typename _Iterator >`
`reverse_iterator< _Iterator > std::operator+ (typename reverse_iterator< _Iterator >::difference_type __n, const reverse_iterator< _Iterator > & __x)`
- `template<typename _Iterator, typename _Container >`
`__normal_iterator< _Iterator, _Container > __gnu_cxx::operator+ (typename __normal_iterator< _Iterator, _Container >::difference_type __n, const __normal_iterator< _Iterator, _Container > & __i) noexcept`
- `template<typename _Iterator >`
`move_iterator< _Iterator > std::operator+ (typename move_iterator< _Iterator >::difference_type __n, const move_iterator< _Iterator > & __x)`
- `template<typename _Iterator >`
`auto std::operator- (const reverse_iterator< _Iterator > & __x, const reverse_iterator< _Iterator > & __y) -> decltype(__x.base()-__y.base())`
- `template<typename _IteratorL, typename _IteratorR >`
`auto std::operator- (const reverse_iterator< _IteratorL > & __x, const reverse_iterator< _IteratorR > & __y) -> decltype(__y.base()-__x.base())`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`auto __gnu_cxx::operator- (const __normal_iterator< _IteratorL, _Container > & __lhs, const __normal_iterator< _IteratorR, _Container > & __rhs) noexcept -> decltype(__lhs.base()-__rhs.base())`

[illegible]

- `template<typename _IteratorL, typename _IteratorR >`
`bool std::operator> (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`bool __gnu_cxx::operator> (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator< _IteratorR, _Container > &__rhs) noexcept`
- `template<typename _Iterator, typename _Container >`
`bool __gnu_cxx::operator> (const __normal_iterator< _Iterator, _Container > &__lhs, const __normal_iterator< _Iterator, _Container > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR >`
`bool std::operator> (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _Iterator >`
`bool std::operator> (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`
- `template<typename _Iterator >`
`bool std::operator>= (const reverse_iterator< _Iterator > &__x, const reverse_iterator< _Iterator > &__y)`
- `template<typename _IteratorL, typename _IteratorR >`
`bool std::operator>= (const reverse_iterator< _IteratorL > &__x, const reverse_iterator< _IteratorR > &__y)`
- `template<typename _IteratorL, typename _IteratorR, typename _Container >`
`bool __gnu_cxx::operator>= (const __normal_iterator< _IteratorL, _Container > &__lhs, const __normal_iterator< _IteratorR, _Container > &__rhs) noexcept`
- `template<typename _Iterator, typename _Container >`
`bool __gnu_cxx::operator>= (const __normal_iterator< _Iterator, _Container > &__lhs, const __normal_iterator< _Iterator, _Container > &__rhs) noexcept`
- `template<typename _IteratorL, typename _IteratorR >`
`bool std::operator>= (const move_iterator< _IteratorL > &__x, const move_iterator< _IteratorR > &__y)`
- `template<typename _Iterator >`
`bool std::operator>= (const move_iterator< _Iterator > &__x, const move_iterator< _Iterator > &__y)`

6.576.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iterator>`.

This file implements `reverse_iterator`, `back_insert_iterator`, `front_insert_iterator`, `insert_iterator`, `__normal_iterator`, and their supporting functions and overloaded operators.

6.577 `stl_iterator.h` File Reference

Namespaces

- [`__gnu_debug`](#)

Functions

- `template<typename _Iterator >`
`auto __gnu_debug::__base (const std::reverse_iterator< _Iterator > &__it) -> decltype(std::__make_reverse_iterator(__base(__it.base())))`
- `template<typename _Iterator >`
`auto __gnu_debug::__base (const std::move_iterator< _Iterator > &__it) -> decltype(std::make_move_iterator(__base(__it.base())))`
- `template<typename _Iterator >`
`_Distance_traits< _Iterator >::__type __gnu_debug::__get_distance (const std::reverse_iterator< _Iterator > &__first, const std::reverse_iterator< _Iterator > &__last)`
- `template<typename _Iterator >`
`_Distance_traits< _Iterator >::__type __gnu_debug::__get_distance (const std::move_iterator< _Iterator > &__first, const std::move_iterator< _Iterator > &__last)`
- `template<typename _Iterator >`
`auto __gnu_debug::__unsafe (const std::reverse_iterator< _Iterator > &__it) -> decltype(std::__make_reverse_iterator(__unsafe(__it.base())))`
- `template<typename _Iterator >`
`auto __gnu_debug::__unsafe (const std::move_iterator< _Iterator > &__it) -> decltype(std::make_move_iterator(__unsafe(__it.base())))`
- `template<typename _Iterator >`
`bool __gnu_debug::__valid_range (const std::reverse_iterator< _Iterator > &__first, const std::reverse_iterator< _Iterator > &__last, typename _Distance_traits< _Iterator >::__type &__dist)`
- `template<typename _Iterator >`
`bool __gnu_debug::__valid_range (const std::move_iterator< _Iterator > &__first, const std::move_iterator< _Iterator > &__last, typename _Distance_traits< _Iterator >::__type &__dist)`

6.577.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.578 `std_iterator_base_funcs.h` File Reference

Classes

- struct [std::_List_const_iterator](#)< _Tp >
- struct [std::_List_iterator](#)< _Tp >

Namespaces

- [std](#)

Functions

- `template<typename _InputIterator, typename _Distance>`
`void std::advance (_InputIterator &__i, _Distance __n, input_iterator_tag)`
- `template<typename _BidirectionalIterator, typename _Distance>`
`void std::advance (_BidirectionalIterator &__i, _Distance __n, bidirectional_iterator_tag)`
- `template<typename _RandomAccessIterator, typename _Distance>`
`void std::advance (_RandomAccessIterator &__i, _Distance __n, random_access_iterator_tag)`
- `template<typename _InputIterator>`
`iterator_traits< _InputIterator >::difference_type std::distance (_InputIterator __first, _InputIterator __last, input_iterator_tag)`
- `template<typename _RandomAccessIterator>`
`iterator_traits< _RandomAccessIterator >::difference_type std::distance (_RandomAccessIterator __first, ↵
↵ _RandomAccessIterator __last, random_access_iterator_tag)`
- `template<typename _InputIterator, typename _Distance>`
`void std::advance (_InputIterator &__i, _Distance __n)`
- `template<typename _InputIterator>`
`iterator_traits< _InputIterator >::difference_type std::distance (_InputIterator __first, _InputIterator __last)`
- `template<typename _ForwardIterator>`
`_ForwardIterator std::next (_ForwardIterator __x, typename iterator_traits< _ForwardIterator >::difference_type
__n=1)`
- `template<typename _BidirectionalIterator>`
`_BidirectionalIterator std::prev (_BidirectionalIterator __x, typename iterator_traits< _BidirectionalIterator >↵
↵::difference_type __n=1)`

6.578.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iterator>`.

This file contains all of the general iterator-related utility functions, such as `distance()` and `advance()`.

6.579 `std_iterator_base_types.h` File Reference

Classes

- `struct std::iterator_traits< _Iterator, typename >`
- `struct std::bidirectional_iterator_tag`
- `struct std::forward_iterator_tag`
- `struct std::input_iterator_tag`
- `struct std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >`
- `struct std::iterator_traits< _Tp * >`
- `struct std::iterator_traits< const _Tp * >`
- `struct std::output_iterator_tag`
- `struct std::random_access_iterator_tag`

Namespaces

- `std`

Typedefs

- `template<typename _InIter >`
`using std::RequireInputIter = typename enable_if< is_convertible< typename iterator_traits< _InIter >::iterator_category, input_iterator_tag >::value >::type`

Functions

- `template<typename _Iter >`
`iterator_traits< _Iter >::iterator_category std::__iterator_category (const _Iter &)`

6.579.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iterator>`.

This file contains all of the general iterator-related utility types, such as `iterator_traits` and `struct iterator`.

6.580 `std::list.h` File Reference

Classes

- `struct std::__detail::List_node_base`
- `class std::List_base< _Tp, _Alloc >`
- `struct std::List_const_iterator< _Tp >`
- `struct std::List_iterator< _Tp >`
- `struct std::List_node< _Tp >`
- `class std::list< _Tp, _Alloc >`

Namespaces

- `std`
- `std::__detail`

Functions

- `template<typename _Val >`
`bool std::operator!= (const List_iterator< _Val > &__x, const List_const_iterator< _Val > &__y) noexcept`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator!= (const list< _Tp, _Alloc > &__x, const list< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator< (const list< _Tp, _Alloc > &__x, const list< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator<= (const list< _Tp, _Alloc > &__x, const list< _Tp, _Alloc > &__y)`
- `template<typename _Val >`
`bool std::operator== (const List_iterator< _Val > &__x, const List_const_iterator< _Val > &__y) noexcept`
- `template<typename _Tp, typename _Alloc >`
`_GLIBCXX_END_NAMESPACE_CXX11 bool std::operator== (const list< _Tp, _Alloc > &__x, const list< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator> (const list< _Tp, _Alloc > &__x, const list< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator>= (const list< _Tp, _Alloc > &__x, const list< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`void std::swap (list< _Tp, _Alloc > &__x, list< _Tp, _Alloc > &__y) noexcept(/*conditional */)`

6.580.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<list>`.

6.581 `std_map.h` File Reference

Classes

- class `std::map<_Key, _Tp, _Compare, _Alloc>`

Namespaces

- `std`

Functions

- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc>`
`bool std::operator!= (const map< _Key, _Tp, _Compare, _Alloc> &__x, const map< _Key, _Tp, _Compare, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc>`
`bool std::operator< (const map< _Key, _Tp, _Compare, _Alloc> &__x, const map< _Key, _Tp, _Compare, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc>`
`bool std::operator<= (const map< _Key, _Tp, _Compare, _Alloc> &__x, const map< _Key, _Tp, _Compare, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc>`
`bool std::operator== (const map< _Key, _Tp, _Compare, _Alloc> &__x, const map< _Key, _Tp, _Compare, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc>`
`bool std::operator> (const map< _Key, _Tp, _Compare, _Alloc> &__x, const map< _Key, _Tp, _Compare, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc>`
`bool std::operator>= (const map< _Key, _Tp, _Compare, _Alloc> &__x, const map< _Key, _Tp, _Compare, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc>`
`void std::swap (map< _Key, _Tp, _Compare, _Alloc> &__x, map< _Key, _Tp, _Compare, _Alloc> &__y)`
`noexcept(/*conditional */)`

6.581.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<map>`.

6.582 `std_multimap.h` File Reference

Classes

- class `std::multimap<_Key, _Tp, _Compare, _Alloc>`

Namespaces

- `std`

Functions

- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc>`
`bool std::operator!= (const multimap<_Key, _Tp, _Compare, _Alloc> &__x, const multimap<_Key, _Tp, _Compare, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc>`
`bool std::operator< (const multimap<_Key, _Tp, _Compare, _Alloc> &__x, const multimap<_Key, _Tp, _Compare, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc>`
`bool std::operator<= (const multimap<_Key, _Tp, _Compare, _Alloc> &__x, const multimap<_Key, _Tp, _Compare, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc>`
`bool std::operator== (const multimap<_Key, _Tp, _Compare, _Alloc> &__x, const multimap<_Key, _Tp, _Compare, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc>`
`bool std::operator> (const multimap<_Key, _Tp, _Compare, _Alloc> &__x, const multimap<_Key, _Tp, _Compare, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc>`
`bool std::operator>= (const multimap<_Key, _Tp, _Compare, _Alloc> &__x, const multimap<_Key, _Tp, _Compare, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Compare, typename _Alloc>`
`void std::swap (multimap<_Key, _Tp, _Compare, _Alloc> &__x, multimap<_Key, _Tp, _Compare, _Alloc> &__y) noexcept(/*conditional */)`

6.582.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<map>`.

6.583 `std_multiset.h` File Reference

Classes

- class `std::multiset<_Key, _Compare, _Alloc>`

Namespaces

- [std](#)

Functions

- `template<typename _Key , typename _Compare , typename _Alloc >`
`bool std::operator!= (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key , typename _Compare , typename _Alloc >`
`bool std::operator< (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key , typename _Compare , typename _Alloc >`
`bool std::operator<= (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key , typename _Compare , typename _Alloc >`
`bool std::operator== (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key , typename _Compare , typename _Alloc >`
`bool std::operator> (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key , typename _Compare , typename _Alloc >`
`bool std::operator>= (const multiset< _Key, _Compare, _Alloc > &__x, const multiset< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key , typename _Compare , typename _Alloc >`
`void std::swap (multiset< _Key, _Compare, _Alloc > &__x, multiset< _Key, _Compare, _Alloc > &__y) noexcept(/*conditional */)`

6.583.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<set>`.

6.584 `stl_numeric.h` File Reference

Namespaces

- [std](#)

Functions

- `template<typename _InputIterator, typename _Tp >`
`_Tp std::accumulate` (`_InputIterator __first`, `_InputIterator __last`, `_Tp __init`)
- `template<typename _InputIterator, typename _Tp, typename _BinaryOperation >`
`_Tp std::accumulate` (`_InputIterator __first`, `_InputIterator __last`, `_Tp __init`, `_BinaryOperation __binary_op`)
- `template<typename _InputIterator, typename _OutputIterator >`
`_OutputIterator std::adjacent_difference` (`_InputIterator __first`, `_InputIterator __last`, `_OutputIterator __result`)
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::adjacent_difference` (`_InputIterator __first`, `_InputIterator __last`, `_OutputIterator __result`, `_BinaryOperation __binary_op`)
- `template<typename _InputIterator1, typename _InputIterator2, typename _Tp >`
`_Tp std::inner_product` (`_InputIterator1 __first1`, `_InputIterator1 __last1`, `_InputIterator2 __first2`, `_Tp __init`)
- `template<typename _InputIterator1, typename _InputIterator2, typename _Tp, typename _BinaryOperation1, typename _BinaryOperation2 >`
`_Tp std::inner_product` (`_InputIterator1 __first1`, `_InputIterator1 __last1`, `_InputIterator2 __first2`, `_Tp __init`, `_BinaryOperation1 __binary_op1`, `_BinaryOperation2 __binary_op2`)
- `template<typename _ForwardIterator, typename _Tp >`
`void std::iota` (`_ForwardIterator __first`, `_ForwardIterator __last`, `_Tp __value`)
- `template<typename _InputIterator, typename _OutputIterator >`
`_OutputIterator std::partial_sum` (`_InputIterator __first`, `_InputIterator __last`, `_OutputIterator __result`)
- `template<typename _InputIterator, typename _OutputIterator, typename _BinaryOperation >`
`_OutputIterator std::partial_sum` (`_InputIterator __first`, `_InputIterator __last`, `_OutputIterator __result`, `_BinaryOperation __binary_op`)

6.584.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<numeric>`.

6.585 `std::pair.h` File Reference

Classes

- struct `std::pair<_T1, _T2 >`
- struct `std::piecewise_construct_t`
- class `std::tuple<_Elements >`

Namespaces

- `std`

Functions

- `template<typename _T1 , typename _T2 >`
`constexpr pair< typename __decay_and_strip< _T1 >::__type, typename __decay_and_strip< _T2 >::__type`
`> std::make_pair (_T1 && __x, _T2 && __y)`
- `template<typename _T1 , typename _T2 >`
`constexpr bool std::operator!= (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename _T1 , typename _T2 >`
`constexpr bool std::operator< (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename _T1 , typename _T2 >`
`constexpr bool std::operator<= (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename _T1 , typename _T2 >`
`constexpr bool std::operator== (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename _T1 , typename _T2 >`
`constexpr bool std::operator> (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename _T1 , typename _T2 >`
`constexpr bool std::operator>= (const pair< _T1, _T2 > &__x, const pair< _T1, _T2 > &__y)`
- `template<typename _T1 , typename _T2 >`
`void std::swap (pair< _T1, _T2 > &__x, pair< _T1, _T2 > &__y) noexcept(noexcept(__x.swap(__y)))`

Variables

- `constexpr piecewise_construct_t std::piecewise_construct`

6.585.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<utility>`.

6.586 `std::queue.h` File Reference

Classes

- class [std::priority_queue< _Tp, _Sequence, _Compare >](#)
- class [std::queue< _Tp, _Sequence >](#)

Namespaces

- [std](#)

Functions

- `template<typename _Tp, typename _Seq >`
`bool std::operator!= (const queue< _Tp, _Seq > &__x, const queue< _Tp, _Seq > &__y)`
- `template<typename _Tp, typename _Seq >`
`bool std::operator< (const queue< _Tp, _Seq > &__x, const queue< _Tp, _Seq > &__y)`
- `template<typename _Tp, typename _Seq >`
`bool std::operator<= (const queue< _Tp, _Seq > &__x, const queue< _Tp, _Seq > &__y)`
- `template<typename _Tp, typename _Seq >`
`bool std::operator== (const queue< _Tp, _Seq > &__x, const queue< _Tp, _Seq > &__y)`
- `template<typename _Tp, typename _Seq >`
`bool std::operator> (const queue< _Tp, _Seq > &__x, const queue< _Tp, _Seq > &__y)`
- `template<typename _Tp, typename _Seq >`
`bool std::operator>= (const queue< _Tp, _Seq > &__x, const queue< _Tp, _Seq > &__y)`
- `template<typename _Tp, typename _Seq >`
`void std::swap (queue< _Tp, _Seq > &__x, queue< _Tp, _Seq > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Tp, typename _Sequence, typename _Compare >`
`void std::swap (priority_queue< _Tp, _Sequence, _Compare > &__x, priority_queue< _Tp, _Sequence, _Compare > &__y) noexcept(noexcept(__x.swap(__y)))`

6.586.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<queue>`.

6.587 `std_raw_storage_iter.h` File Reference

Classes

- `class std::raw_storage_iterator< _OutputIterator, _Tp >`

Namespaces

- [std](#)

6.587.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

6.588 `std_relops.h` File Reference

Namespaces

- [std](#)
- [std::rel_ops](#)

Functions

- `template<class _Tp >`
`bool std::rel_ops::operator!= (const _Tp &__x, const _Tp &__y)`
- `template<class _Tp >`
`bool std::rel_ops::operator<= (const _Tp &__x, const _Tp &__y)`
- `template<class _Tp >`
`bool std::rel_ops::operator> (const _Tp &__x, const _Tp &__y)`
- `template<class _Tp >`
`bool std::rel_ops::operator>= (const _Tp &__x, const _Tp &__y)`

6.588.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<utility>`.

Inclusion of this file has been removed from all of the other STL headers for safety reasons, except `std_`↵
`utility.h`. For more information, see the thread of about twenty messages starting with <http://gcc.gnu.org/ml/libstdc++/2001-01/msg00223.html>, or [http://gcc.gnu.org/onlinedocs/libstdc++/faq.](http://gcc.gnu.org/onlinedocs/libstdc++/faq.html#faq.ambiguous_overloads)↵
[html#faq.ambiguous_overloads](http://gcc.gnu.org/onlinedocs/libstdc++/faq.html#faq.ambiguous_overloads)

Short summary: the `rel_ops` operators should be avoided for the present.

6.589 std_set.h File Reference

Classes

- class `std::set< _Key, _Compare, _Alloc >`

Namespaces

- `std`

Functions

- `template<typename _Key , typename _Compare , typename _Alloc >`
`bool std::operator!= (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key , typename _Compare , typename _Alloc >`
`bool std::operator< (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key , typename _Compare , typename _Alloc >`
`bool std::operator<= (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key , typename _Compare , typename _Alloc >`
`bool std::operator== (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key , typename _Compare , typename _Alloc >`
`bool std::operator> (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key , typename _Compare , typename _Alloc >`
`bool std::operator>= (const set< _Key, _Compare, _Alloc > &__x, const set< _Key, _Compare, _Alloc > &__y)`
- `template<typename _Key , typename _Compare , typename _Alloc >`
`void std::swap (set< _Key, _Compare, _Alloc > &__x, set< _Key, _Compare, _Alloc > &__y) noexcept(/*conditional */)`

6.589.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<set>`.

6.590 `std_stack.h` File Reference

Classes

- class `std::stack<_Tp, _Sequence>`

Namespaces

- `std`

Functions

- `template<typename _Tp, typename _Seq>`
`bool std::operator!= (const stack<_Tp, _Seq> &__x, const stack<_Tp, _Seq> &__y)`
- `template<typename _Tp, typename _Seq>`
`bool std::operator< (const stack<_Tp, _Seq> &__x, const stack<_Tp, _Seq> &__y)`
- `template<typename _Tp, typename _Seq>`
`bool std::operator<= (const stack<_Tp, _Seq> &__x, const stack<_Tp, _Seq> &__y)`
- `template<typename _Tp, typename _Seq>`
`bool std::operator== (const stack<_Tp, _Seq> &__x, const stack<_Tp, _Seq> &__y)`
- `template<typename _Tp, typename _Seq>`
`bool std::operator> (const stack<_Tp, _Seq> &__x, const stack<_Tp, _Seq> &__y)`
- `template<typename _Tp, typename _Seq>`
`bool std::operator>= (const stack<_Tp, _Seq> &__x, const stack<_Tp, _Seq> &__y)`
- `template<typename _Tp, typename _Seq>`
`void std::swap (stack<_Tp, _Seq> &__x, stack<_Tp, _Seq> &__y) noexcept(noexcept(__x.swap(__y)))`

6.590.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<stack>`.

6.591 `std_tempbuf.h` File Reference

Classes

- class `std::_Temporary_buffer<_ForwardIterator, _Tp>`

Namespaces

- [std](#)

Functions

- `template<typename _Pointer, typename _ForwardIterator >
void std::__uninitialized_construct_buf (_Pointer __first, _Pointer __last, _ForwardIterator __seed)`
- `template<typename _Tp >
pair< _Tp *, ptrdiff_t > std::get_temporary_buffer (ptrdiff_t __len) noexcept`
- `template<typename _Tp >
void std::return_temporary_buffer (_Tp *__p)`

6.591.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

6.592 `stl_tree.h` File Reference

Namespaces

- [std](#)

Macros

- `#define __cpp_lib_generic_associative_lookup`

Enumerations

- `enum _Rb_tree_color { _S_red, _S_black }`

Functions

- unsigned int **std::Rb_tree_black_count** (const `_Rb_tree_node_base` *__node, const `_Rb_tree_node_base` *__root) throw ()
- `_Rb_tree_node_base` * **std::Rb_tree_decrement** (`_Rb_tree_node_base` *__x) throw ()
- const `_Rb_tree_node_base` * **std::Rb_tree_decrement** (const `_Rb_tree_node_base` *__x) throw ()
- `_Rb_tree_node_base` * **std::Rb_tree_increment** (`_Rb_tree_node_base` *__x) throw ()
- const `_Rb_tree_node_base` * **std::Rb_tree_increment** (const `_Rb_tree_node_base` *__x) throw ()
- void **std::Rb_tree_insert_and_rebalance** (const bool __insert_left, `_Rb_tree_node_base` *__x, `_Rb_tree_node_base` *__p, `_Rb_tree_node_base` &__header) throw ()
- `_Rb_tree_node_base` * **std::Rb_tree_rebalance_for_erase** (`_Rb_tree_node_base` *const __z, `_Rb_tree_node_base` &__header) throw ()
- template<typename `_Val` >
bool **std::operator!=** (const `_Rb_tree_iterator`< `_Val` > &__x, const `_Rb_tree_const_iterator`< `_Val` > &__y) noexcept
- template<typename `_Key`, typename `_Val`, typename `_KeyOfValue`, typename `_Compare`, typename `_Alloc` >
bool **std::operator!=** (const `_Rb_tree`< `_Key`, `_Val`, `_KeyOfValue`, `_Compare`, `_Alloc` > &__x, const `_Rb_tree`< `_Key`, `_Val`, `_KeyOfValue`, `_Compare`, `_Alloc` > &__y)
- template<typename `_Key`, typename `_Val`, typename `_KeyOfValue`, typename `_Compare`, typename `_Alloc` >
bool **std::operator<** (const `_Rb_tree`< `_Key`, `_Val`, `_KeyOfValue`, `_Compare`, `_Alloc` > &__x, const `_Rb_tree`< `_Key`, `_Val`, `_KeyOfValue`, `_Compare`, `_Alloc` > &__y)
- template<typename `_Key`, typename `_Val`, typename `_KeyOfValue`, typename `_Compare`, typename `_Alloc` >
bool **std::operator<=** (const `_Rb_tree`< `_Key`, `_Val`, `_KeyOfValue`, `_Compare`, `_Alloc` > &__x, const `_Rb_tree`< `_Key`, `_Val`, `_KeyOfValue`, `_Compare`, `_Alloc` > &__y)
- template<typename `_Val` >
bool **std::operator==** (const `_Rb_tree_iterator`< `_Val` > &__x, const `_Rb_tree_const_iterator`< `_Val` > &__y) noexcept
- template<typename `_Key`, typename `_Val`, typename `_KeyOfValue`, typename `_Compare`, typename `_Alloc` >
bool **std::operator==** (const `_Rb_tree`< `_Key`, `_Val`, `_KeyOfValue`, `_Compare`, `_Alloc` > &__x, const `_Rb_tree`< `_Key`, `_Val`, `_KeyOfValue`, `_Compare`, `_Alloc` > &__y)
- template<typename `_Key`, typename `_Val`, typename `_KeyOfValue`, typename `_Compare`, typename `_Alloc` >
bool **std::operator>** (const `_Rb_tree`< `_Key`, `_Val`, `_KeyOfValue`, `_Compare`, `_Alloc` > &__x, const `_Rb_tree`< `_Key`, `_Val`, `_KeyOfValue`, `_Compare`, `_Alloc` > &__y)
- template<typename `_Key`, typename `_Val`, typename `_KeyOfValue`, typename `_Compare`, typename `_Alloc` >
bool **std::operator>=** (const `_Rb_tree`< `_Key`, `_Val`, `_KeyOfValue`, `_Compare`, `_Alloc` > &__x, const `_Rb_tree`< `_Key`, `_Val`, `_KeyOfValue`, `_Compare`, `_Alloc` > &__y)
- template<typename `_Key`, typename `_Val`, typename `_KeyOfValue`, typename `_Compare`, typename `_Alloc` >
void **std::swap** (`_Rb_tree`< `_Key`, `_Val`, `_KeyOfValue`, `_Compare`, `_Alloc` > &__x, `_Rb_tree`< `_Key`, `_Val`, `_KeyOfValue`, `_Compare`, `_Alloc` > &__y)

6.592.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<map>` or `<set>`.

6.593 `std_uninitialized.h` File Reference

Namespaces

- [std](#)

Functions

- `template<typename _InputIterator, typename _ForwardIterator, typename _Allocator >`
`_ForwardIterator std:: uninitialized_copy_a (_InputIterator __first, _InputIterator __last, _ForwardIterator __result, _Allocator & __alloc)`
- `template<typename _InputIterator, typename _ForwardIterator, typename _Tp >`
`_ForwardIterator std:: uninitialized_copy_a (_InputIterator __first, _InputIterator __last, _ForwardIterator __result, allocator< _Tp > &)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _ForwardIterator, typename _Allocator >`
`_ForwardIterator std:: uninitialized_copy_move (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _ForwardIterator __result, _Allocator & __alloc)`
- `template<typename _InputIterator, typename _Size, typename _ForwardIterator >`
`_ForwardIterator std:: uninitialized_copy_n (_InputIterator __first, _Size __n, _ForwardIterator __result, input_iterator_tag)`
- `template<typename _RandomAccessIterator, typename _Size, typename _ForwardIterator >`
`_ForwardIterator std:: uninitialized_copy_n (_RandomAccessIterator __first, _Size __n, _ForwardIterator __result, random_access_iterator_tag)`
- `template<typename _ForwardIterator >`
`void std:: uninitialized_default (_ForwardIterator __first, _ForwardIterator __last)`
- `template<typename _ForwardIterator, typename _Allocator >`
`void std:: uninitialized_default_a (_ForwardIterator __first, _ForwardIterator __last, _Allocator & __alloc)`
- `template<typename _ForwardIterator, typename _Tp >`
`void std:: uninitialized_default_a (_ForwardIterator __first, _ForwardIterator __last, allocator< _Tp > &)`
- `template<typename _ForwardIterator, typename _Size >`
`_ForwardIterator std:: uninitialized_default_n (_ForwardIterator __first, _Size __n)`
- `template<typename _ForwardIterator, typename _Size, typename _Allocator >`
`_ForwardIterator std:: uninitialized_default_n_a (_ForwardIterator __first, _Size __n, _Allocator & __alloc)`
- `template<typename _ForwardIterator, typename _Size, typename _Tp >`
`_ForwardIterator std:: uninitialized_default_n_a (_ForwardIterator __first, _Size __n, allocator< _Tp > &)`
- `template<typename _ForwardIterator, typename _Tp, typename _Allocator >`
`void std:: uninitialized_fill_a (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __x, _Allocator & __alloc)`
- `template<typename _ForwardIterator, typename _Tp, typename _Tp2 >`
`void std:: uninitialized_fill_a (_ForwardIterator __first, _ForwardIterator __last, const _Tp & __x, allocator< _Tp2 > &)`
- `template<typename _ForwardIterator, typename _Tp, typename _InputIterator, typename _Allocator >`
`_ForwardIterator std:: uninitialized_fill_move (_ForwardIterator __result, _ForwardIterator __mid, const _Tp & __x, _InputIterator __first, _InputIterator __last, _Allocator & __alloc)`
- `template<typename _ForwardIterator, typename _Size, typename _Tp, typename _Allocator >`
`_ForwardIterator std:: uninitialized_fill_n_a (_ForwardIterator __first, _Size __n, const _Tp & __x, _Allocator & __alloc)`
- `template<typename _ForwardIterator, typename _Size, typename _Tp, typename _Tp2 >`
`_ForwardIterator std:: uninitialized_fill_n_a (_ForwardIterator __first, _Size __n, const _Tp & __x, allocator< _Tp2 > &)`
- `template<typename _InputIterator, typename _ForwardIterator, typename _Allocator >`
`_ForwardIterator std:: uninitialized_move_a (_InputIterator __first, _InputIterator __last, _ForwardIterator __result, _Allocator & __alloc)`
- `template<typename _InputIterator1, typename _InputIterator2, typename _ForwardIterator, typename _Allocator >`
`_ForwardIterator std:: uninitialized_move_copy (_InputIterator1 __first1, _InputIterator1 __last1, _InputIterator2 __first2, _InputIterator2 __last2, _ForwardIterator __result, _Allocator & __alloc)`
- `template<typename _InputIterator, typename _ForwardIterator, typename _Tp, typename _Allocator >`
`void std:: uninitialized_move_fill (_InputIterator __first1, _InputIterator __last1, _ForwardIterator __first2, _ForwardIterator __last2, const _Tp & __x, _Allocator & __alloc)`

- `template<typename _InputIterator, typename _ForwardIterator, typename _Allocator >`
`_ForwardIterator std::uninitialized_move_if_noexcept_a (_InputIterator __first, _InputIterator __last, _↔`
`ForwardIterator __result, _Allocator &__alloc)`
- `template<typename _InputIterator, typename _ForwardIterator >`
`_ForwardIterator std::uninitialized_copy (_InputIterator __first, _InputIterator __last, _ForwardIterator __result)`
- `template<typename _InputIterator, typename _Size, typename _ForwardIterator >`
`_ForwardIterator std::uninitialized_copy_n (_InputIterator __first, _Size __n, _ForwardIterator __result)`
- `template<typename _ForwardIterator, typename _Tp >`
`void std::uninitialized_fill (_ForwardIterator __first, _ForwardIterator __last, const _Tp &__x)`
- `template<typename _ForwardIterator, typename _Size, typename _Tp >`
`_ForwardIterator std::uninitialized_fill_n (_ForwardIterator __first, _Size __n, const _Tp &__x)`

6.593.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

6.594 `std_vector.h` File Reference

Classes

- struct `std::_Vector_base< _Tp, _Alloc >`
- class `std::vector< _Tp, _Alloc >`

Namespaces

- `std`

Functions

- `template<typename _Tp, typename _Alloc >`
`bool std::operator!= (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator< (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator<= (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator== (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator> (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`bool std::operator>= (const vector< _Tp, _Alloc > &__x, const vector< _Tp, _Alloc > &__y)`
- `template<typename _Tp, typename _Alloc >`
`void std::swap (vector< _Tp, _Alloc > &__x, vector< _Tp, _Alloc > &__y) noexcept(/*conditional */)`

6.594.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<vector>`.

6.595 `stream_iterator.h` File Reference

Classes

- class [std::istream_iterator<_Tp, _CharT, _Traits, _Dist>](#)
- class [std::ostream_iterator<_Tp, _CharT, _Traits>](#)

Namespaces

- [std](#)

Functions

- `template<class _Tp, class _CharT, class _Traits, class _Dist>
bool std::operator!= (const istream_iterator<_Tp, _CharT, _Traits, _Dist> &__x, const istream_iterator<_Tp, _CharT, _Traits, _Dist> &__y)`
- `template<typename _Tp, typename _CharT, typename _Traits, typename _Dist>
bool std::operator== (const istream_iterator<_Tp, _CharT, _Traits, _Dist> &__x, const istream_iterator<_Tp, _CharT, _Traits, _Dist> &__y)`

6.595.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iterator>`.

6.596 `streambuf` File Reference

Classes

- class [std::basic_streambuf<_CharT, _Traits>](#)

Namespaces

- [std](#)

Macros

- `#define _GLIBXX_STREAMBUF`

Functions

- `template<typename _CharT, typename _Traits >`
`streamsize std::__copy_streambufs_eof (basic_streambuf< _CharT, _Traits > *, basic_streambuf< _CharT, _Traits > *, bool &)`
- `template<>`
`streamsize std::__copy_streambufs_eof (basic_streambuf< char > *__sbin, basic_streambuf< char > *__sout, bool &__ineof)`
- `template<>`
`streamsize std::__copy_streambufs_eof (basic_streambuf< wchar_t > *__sbin, basic_streambuf< wchar_t > *__sout, bool &__ineof)`

6.596.1 Detailed Description

This is a Standard C++ Library header.

6.597 streambuf.tcc File Reference

Namespaces

- [std](#)

Macros

- `#define _STREAMBUF_TCC`

Functions

- `template<typename _CharT, typename _Traits >`
`streamsize std::__copy_streambufs (basic_streambuf< _CharT, _Traits > *__sbin, basic_streambuf< _CharT, _Traits > *__sout)`
- `template<typename _CharT, typename _Traits >`
`streamsize std::__copy_streambufs_eof (basic_streambuf< _CharT, _Traits > *, basic_streambuf< _CharT, _Traits > *, bool &)`

6.597.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<streambuf>`.

6.598 streambuf_iterator.h File Reference

Classes

- class [std::istreambuf_iterator< _CharT, _Traits >](#)
- class [std::ostreambuf_iterator< _CharT, _Traits >](#)

Namespaces

- [std](#)

Functions

- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, ostreambuf_iterator< _CharT > >::__type std::↵`
`copy_move_a2 (_CharT * __first, _CharT * __last, ostreambuf_iterator< _CharT > __result)`
- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, ostreambuf_iterator< _CharT > >::__type std::↵`
`copy_move_a2 (const _CharT * __first, const _CharT * __last, ostreambuf_iterator< _CharT > __result)`
- `template<bool _IsMove, typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, _CharT * >::__type std::↵`
`copy_move_a2 (istreambuf_iterator< _CharT > __first, istreambuf_iterator< _CharT > __last, _CharT * __result)`
- `template<typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, ostreambuf_iterator< _CharT > >::__type std::↵`
`copy (istreambuf_iterator< _CharT > __first, istreambuf_iterator< _CharT > __last, ostreambuf_iterator< _CharT >`
`__result)`
- `template<typename _CharT >`
`__gnu_cxx::__enable_if< __is_char< _CharT >::__value, istreambuf_iterator< _CharT > >::__type std::↵`
`find (istreambuf_iterator< _CharT > __first, istreambuf_iterator< _CharT > __last, const _CharT & __val)`
- `template<typename _CharT, typename _Traits >`
`bool std::operator!= (const istreambuf_iterator< _CharT, _Traits > & __a, const istreambuf_iterator< _CharT,`
`_Traits > & __b)`
- `template<typename _CharT, typename _Traits >`
`bool std::operator== (const istreambuf_iterator< _CharT, _Traits > & __a, const istreambuf_iterator< _CharT,`
`_Traits > & __b)`

6.598.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<iterator>`.

6.599 string File Reference

Macros

- `#define _GLIBCXX_STRING`

6.599.1 Detailed Description

This is a Standard C++ Library header.

6.600 string File Reference

Classes

- class [__gnu_debug::basic_string<_CharT, _Traits, _Allocator>](#)

Namespaces

- [__gnu_debug](#)

Macros

- `#define _GLIBCXX_DEBUG_STRING`

Typedefs

- typedef basic_string< char > [__gnu_debug::string](#)
- typedef basic_string< wchar_t > [__gnu_debug::wstring](#)

Functions

- template<typename _CharT, typename _Traits, typename _Allocator>
[std::basic_istream<_CharT, _Traits>](#) & [__gnu_debug::getline](#) ([std::basic_istream<_CharT, _Traits>](#) &__is, basic_string<_CharT, _Traits, _Allocator> &__str, _CharT __delim)
- template<typename _CharT, typename _Traits, typename _Allocator>
[std::basic_istream<_CharT, _Traits>](#) & [__gnu_debug::getline](#) ([std::basic_istream<_CharT, _Traits>](#) &__is, basic_string<_CharT, _Traits, _Allocator> &__str)
- template<typename _CharT, typename _Traits, typename _Allocator>
bool [__gnu_debug::operator!=](#) (const basic_string<_CharT, _Traits, _Allocator> &__lhs, const basic_string<_CharT, _Traits, _Allocator> &__rhs)
- template<typename _CharT, typename _Traits, typename _Allocator>
bool [__gnu_debug::operator!=](#) (const _CharT *__lhs, const basic_string<_CharT, _Traits, _Allocator> &__↵
rhs)
- template<typename _CharT, typename _Traits, typename _Allocator>
bool [__gnu_debug::operator!=](#) (const basic_string<_CharT, _Traits, _Allocator> &__lhs, const _CharT *__↵
rhs)
- template<typename _CharT, typename _Traits, typename _Allocator>
basic_string<_CharT, _Traits, _Allocator> [__gnu_debug::operator+](#) (const basic_string<_CharT, _Traits, ↵
_Allocator> &__lhs, const basic_string<_CharT, _Traits, _Allocator> &__rhs)
- template<typename _CharT, typename _Traits, typename _Allocator>
basic_string<_CharT, _Traits, _Allocator> [__gnu_debug::operator+](#) (const _CharT *__lhs, const basic_↵
string<_CharT, _Traits, _Allocator> &__rhs)
- template<typename _CharT, typename _Traits, typename _Allocator>
basic_string<_CharT, _Traits, _Allocator> [__gnu_debug::operator+](#) (_CharT __lhs, const basic_string< ↵
_CharT, _Traits, _Allocator> &__rhs)
- template<typename _CharT, typename _Traits, typename _Allocator>
basic_string<_CharT, _Traits, _Allocator> [__gnu_debug::operator+](#) (const basic_string<_CharT, _Traits, ↵
_Allocator> &__lhs, const _CharT *__rhs)

- `template<typename _CharT, typename _Traits, typename _Allocator >`
`void __gnu_debug::swap (basic_string< _CharT, _Traits, _Allocator > &__lhs, basic_string< _CharT, _Traits, _Allocator > &__rhs)`

6.600.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.601 string File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_STRING`

Typedefs

- `template<typename _CharT, typename _Traits = char_traits<_CharT>>`
`using std::experimental::fundamentals_v2::pmr::basic_string = std::basic_string< _CharT, _Traits, polymorphic_allocator< _CharT >>`
- `typedef basic_string< char > std::experimental::fundamentals_v2::pmr::string`
- `typedef basic_string< char16_t > std::experimental::fundamentals_v2::pmr::u16string`
- `typedef basic_string< char32_t > std::experimental::fundamentals_v2::pmr::u32string`
- `typedef basic_string< wchar_t > std::experimental::fundamentals_v2::pmr::wstring`

Functions

- `template<typename _CharT, typename _Traits, typename _Alloc, typename _Up >`
`void std::experimental::fundamentals_v2::erase (basic_string< _CharT, _Traits, _Alloc > &__cont, const _Up &__value)`
- `template<typename _CharT, typename _Traits, typename _Alloc, typename _Predicate >`
`void std::experimental::fundamentals_v2::erase_if (basic_string< _CharT, _Traits, _Alloc > &__cont, $_↵$ Predicate __pred)`

6.601.1 Detailed Description

This is a TS C++ Library header.

6.602 string_conversions.h File Reference

Namespaces

- [__gnu_cxx](#)

Functions

- `template<typename _TRet , typename _Ret = _TRet, typename _CharT , typename... _Base>
_Ret __gnu_cxx::__stoa (_TRet(*__convf)(const _CharT *, _CharT **, _Base...), const char *__name, const
_CharT *__str, std::size_t *__idx, _Base...__base)`
- `template<typename _String , typename _CharT = typename _String::value_type>
_String __gnu_cxx::__to_xstring (int(*__convf)(_CharT *, std::size_t, const _CharT *, __builtin_va_list), std::size_t __n, const _CharT *__fmt,...)`

6.602.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.603 string_view File Reference

Classes

- class [std::experimental::fundamentals_v1::basic_string_view< _CharT, _Traits >](#)
- struct [std::hash< _Tp >](#)

Namespaces

- [std](#)

Macros

- `#define __cpp_lib_experimental_string_view`
- `#define _GLIBCXX_EXPERIMENTAL_STRING_VIEW`

Typedefs

- `template<typename _Tp >
using std::experimental::fundamentals_v1::__detail::__idt = typename __identity< _Tp >::type`
- `using std::experimental::fundamentals_v1::string_view = basic_string_view< char >`
- `using std::experimental::fundamentals_v1::u16string_view = basic_string_view< char16_t >`
- `using std::experimental::fundamentals_v1::u32string_view = basic_string_view< char32_t >`
- `using std::experimental::fundamentals_v1::wstring_view = basic_string_view< wchar_t >`

Functions

- `template<typename _CharT, typename _Traits >`
`bool std::experimental::fundamentals_v1::operator!= (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`bool std::experimental::fundamentals_v1::operator!= (basic_string_view< _CharT, _Traits > __x, __detail::__idt< basic_string_view< _CharT, _Traits >> __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`bool std::experimental::fundamentals_v1::operator!= (__detail::__idt< basic_string_view< _CharT, _Traits >> __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `constexpr basic_string_view< char > std::experimental::literals::string_view_literals::operator""sv (const char *__str, size_t __len)`
- `constexpr basic_string_view< wchar_t > std::experimental::literals::string_view_literals::operator""sv (const wchar_t *__str, size_t __len)`
- `constexpr basic_string_view< char16_t > std::experimental::literals::string_view_literals::operator""sv (const char16_t *__str, size_t __len)`
- `constexpr basic_string_view< char32_t > std::experimental::literals::string_view_literals::operator""sv (const char32_t *__str, size_t __len)`
- `template<typename _CharT, typename _Traits >`
`bool std::experimental::fundamentals_v1::operator< (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`bool std::experimental::fundamentals_v1::operator< (basic_string_view< _CharT, _Traits > __x, __detail::__idt< basic_string_view< _CharT, _Traits >> __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`bool std::experimental::fundamentals_v1::operator< (__detail::__idt< basic_string_view< _CharT, _Traits >> __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`basic_ostream< _CharT, _Traits > & std::experimental::fundamentals_v1::operator<< (basic_ostream< _CharT, _Traits > & __os, basic_string_view< _CharT, _Traits > __str)`
- `template<typename _CharT, typename _Traits >`
`bool std::experimental::fundamentals_v1::operator<= (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`bool std::experimental::fundamentals_v1::operator<= (basic_string_view< _CharT, _Traits > __x, __detail::__idt< basic_string_view< _CharT, _Traits >> __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`bool std::experimental::fundamentals_v1::operator<= (__detail::__idt< basic_string_view< _CharT, _Traits >> __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`bool std::experimental::fundamentals_v1::operator== (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`bool std::experimental::fundamentals_v1::operator== (basic_string_view< _CharT, _Traits > __x, __detail::__idt< basic_string_view< _CharT, _Traits >> __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`bool std::experimental::fundamentals_v1::operator== (__detail::__idt< basic_string_view< _CharT, _Traits >> __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`bool std::experimental::fundamentals_v1::operator> (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits > __y) noexcept`

- `template<typename _CharT, typename _Traits >`
`bool std::experimental::fundamentals_v1::operator> (basic_string_view< _CharT, _Traits > __x, __detail::__id< basic_string_view< _CharT, _Traits >> __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`bool std::experimental::fundamentals_v1::operator> (__detail::__id< basic_string_view< _CharT, _Traits >> __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`bool std::experimental::fundamentals_v1::operator>= (basic_string_view< _CharT, _Traits > __x, basic_string_view< _CharT, _Traits > __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`bool std::experimental::fundamentals_v1::operator>= (basic_string_view< _CharT, _Traits > __x, __detail::__id< basic_string_view< _CharT, _Traits >> __y) noexcept`
- `template<typename _CharT, typename _Traits >`
`bool std::experimental::fundamentals_v1::operator>= (__detail::__id< basic_string_view< _CharT, _Traits >> __x, basic_string_view< _CharT, _Traits > __y) noexcept`

6.603.1 Detailed Description

This is a TS C++ Library header.

6.604 string_view.tcc File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_STRING_VIEW_TCC`

6.604.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<experimental/string_view>`.

6.605 stringfwd.h File Reference

Classes

- class [std::basic_string< _CharT, _Traits, _Alloc >](#)
- struct [std::char_traits< _CharT >](#)

Namespaces

- [std](#)

Typedefs

- typedef `basic_string< char >` [std::string](#)
- typedef `basic_string< char16_t >` [std::u16string](#)
- typedef `basic_string< char32_t >` [std::u32string](#)
- typedef `basic_string< wchar_t >` [std::wstring](#)

6.605.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<string>`.

6.606 `stringstream` File Reference

Namespaces

- [std](#)

6.606.1 Detailed Description

This is a Standard C++ Library header.

6.607 `synth_access_traits.hpp` File Reference

Classes

- struct [__gnu_pbds::detail::synth_access_traits< Type_Traits, Set, _ATraits >](#)

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_SYNTH_E_ACCESS_TRAITS_C_DEC`
- `#define PB_DS_SYNTH_E_ACCESS_TRAITS_T_DEC`

6.607.1 Detailed Description

Contains an implementation class for a patricia tree.

6.608 system_error File Reference

Classes

- class [std::_V2::error_category](#)
- struct [std::error_code](#)
- struct [std::error_condition](#)
- struct [std::hash< _Tp >](#)
- struct [std::hash< error_code >](#)
- struct [std::is_error_code_enum< _Tp >](#)
- struct [std::is_error_condition_enum< _Tp >](#)
- class [std::system_error](#)

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_SYSTEM_ERROR`

Functions

- `const error_category & std::_V2::generic_category () noexcept`
- `error_code std::make_error_code (errc __e) noexcept`
- `error_condition std::make_error_condition (errc __e) noexcept`
- `bool std::operator!= (const error_code &__lhs, const error_code &__rhs) noexcept`
- `bool std::operator!= (const error_code &__lhs, const error_condition &__rhs) noexcept`
- `bool std::operator!= (const error_condition &__lhs, const error_code &__rhs) noexcept`
- `bool std::operator!= (const error_condition &__lhs, const error_condition &__rhs) noexcept`
- `bool std::operator< (const error_code &__lhs, const error_code &__rhs) noexcept`
- `bool std::operator< (const error_condition &__lhs, const error_condition &__rhs) noexcept`
- `template<typename _CharT, typename _Traits >
basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, const error_←
_code &__e)`
- `bool std::operator== (const error_code &__lhs, const error_code &__rhs) noexcept`
- `bool std::operator== (const error_code &__lhs, const error_condition &__rhs) noexcept`
- `bool std::operator== (const error_condition &__lhs, const error_code &__rhs) noexcept`
- `bool std::operator== (const error_condition &__lhs, const error_condition &__rhs) noexcept`
- `const error_category & std::_V2::system_category () noexcept`

Variables

- `error_code std::make_error_code (errc) noexcept`
- `error_condition std::make_error_condition (errc) noexcept`

6.608.1 Detailed Description

This is a Standard C++ Library header.

6.609 **system_error** File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_SYSTEM_ERROR`

Variables

- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_error_code_enum_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_error_condition_enum_v`

6.609.1 Detailed Description

This is a TS C++ Library header.

6.610 **tag_and_trait.hpp** File Reference

Classes

- `struct __gnu_pbds::associative_tag`
- `struct __gnu_pbds::basic_branch_tag`
- `struct __gnu_pbds::basic_hash_tag`
- `struct __gnu_pbds::basic_invalidation_guarantee`
- `struct __gnu_pbds::binary_heap_tag`
- `struct __gnu_pbds::binomial_heap_tag`
- `struct __gnu_pbds::cc_hash_tag`
- `struct __gnu_pbds::container_tag`
- `struct __gnu_pbds::container_traits< Cntnr >`
- `struct __gnu_pbds::container_traits_base< _Tag >`
- `struct __gnu_pbds::container_traits_base< binary_heap_tag >`
- `struct __gnu_pbds::container_traits_base< binomial_heap_tag >`
- `struct __gnu_pbds::container_traits_base< cc_hash_tag >`
- `struct __gnu_pbds::container_traits_base< gp_hash_tag >`
- `struct __gnu_pbds::container_traits_base< list_update_tag >`
- `struct __gnu_pbds::container_traits_base< ov_tree_tag >`

- [struct __gnu_pbds::container_traits_base< pairing_heap_tag >](#)
- [struct __gnu_pbds::container_traits_base< pat_trie_tag >](#)
- [struct __gnu_pbds::container_traits_base< rb_tree_tag >](#)
- [struct __gnu_pbds::container_traits_base< rc_binomial_heap_tag >](#)
- [struct __gnu_pbds::container_traits_base< splay_tree_tag >](#)
- [struct __gnu_pbds::container_traits_base< thin_heap_tag >](#)
- [struct __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, Tag, Policy_Tl >](#)
- [struct __gnu_pbds::gp_hash_tag](#)
- [struct __gnu_pbds::list_update_tag](#)
- [struct __gnu_pbds::null_node_update< _Tp1, _Tp2, _Tp3, _Tp4 >](#)
- [struct __gnu_pbds::null_type](#)
- [struct __gnu_pbds::ov_tree_tag](#)
- [struct __gnu_pbds::pairing_heap_tag](#)
- [struct __gnu_pbds::pat_trie_tag](#)
- [struct __gnu_pbds::point_invalidation_guarantee](#)
- [struct __gnu_pbds::priority_queue_tag](#)
- [struct __gnu_pbds::range_invalidation_guarantee](#)
- [struct __gnu_pbds::rb_tree_tag](#)
- [struct __gnu_pbds::rc_binomial_heap_tag](#)
- [struct __gnu_pbds::sequence_tag](#)
- [struct __gnu_pbds::splay_tree_tag](#)
- [struct __gnu_pbds::string_tag](#)
- [struct __gnu_pbds::thin_heap_tag](#)
- [struct __gnu_pbds::tree_tag](#)
- [struct __gnu_pbds::trie_tag](#)
- [struct __gnu_pbds::trivial_iterator_tag](#)

Namespaces

- [__gnu_pbds](#)

Typedefs

- [typedef void __gnu_pbds::trivial_iterator_difference_type](#)

6.610.1 Detailed Description

Contains tags and traits, e.g., ones describing underlying data structures.

6.611 tags.h File Reference

Classes

- struct [__gnu_parallel::balanced_quicksort_tag](#)
- struct [__gnu_parallel::balanced_tag](#)
- struct [__gnu_parallel::constant_size_blocks_tag](#)
- struct [__gnu_parallel::default_parallel_tag](#)
- struct [__gnu_parallel::equal_split_tag](#)
- struct [__gnu_parallel::exact_tag](#)
- struct [__gnu_parallel::find_tag](#)
- struct [__gnu_parallel::growing_blocks_tag](#)
- struct [__gnu_parallel::multiway_mergesort_exact_tag](#)
- struct [__gnu_parallel::multiway_mergesort_sampling_tag](#)
- struct [__gnu_parallel::multiway_mergesort_tag](#)
- struct [__gnu_parallel::omp_loop_static_tag](#)
- struct [__gnu_parallel::omp_loop_tag](#)
- struct [__gnu_parallel::parallel_tag](#)
- struct [__gnu_parallel::quicksort_tag](#)
- struct [__gnu_parallel::sampling_tag](#)
- struct [__gnu_parallel::sequential_tag](#)
- struct [__gnu_parallel::unbalanced_tag](#)

Namespaces

- [__gnu_parallel](#)

6.611.1 Detailed Description

Tags for compile-time selection. This file is a GNU parallel extension to the Standard C++ Library.

6.612 tgmth.h File Reference

Macros

- `#define _GLIBCXX_TGMATH_H`

6.612.1 Detailed Description

This is a Standard C++ Library header.

6.613 thin_heap.hpp File Reference

Classes

- class [__gnu_pbds::detail::thin_heap< Value_Type, Cmp_Fn, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_ASSERT_AUX_NULL(X)`
- `#define PB_DS_ASSERT_NODE_CONSISTENT(_Node, _Bool)`
- `#define PB_DS_BASE_T_P`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`

Enumerations

- `enum { num_distinct_rank_bounds }`

Variables

- `static const std::size_t __gnu_pbds::detail::g_a_rank_bounds [num_distinct_rank_bounds]`

6.613.1 Detailed Description

Contains an implementation class for a thin heap.

6.614 thread File Reference

Classes

- struct [std::hash< thread::id >](#)
- class [std::thread](#)
- class [std::thread::id](#)

Namespaces

- [std](#)
- [std::this_thread](#)

Macros

- `#define _GLIBCXX_THREAD`

Functions

- void **std::this_thread::__sleep_for** (chrono::seconds, chrono::nanoseconds)
- thread::id **std::this_thread::get_id** () noexcept
- bool **std::operator!=** (thread::id __x, thread::id __y) noexcept
- template<class _CharT, class _Traits >
basic_ostream< _CharT, _Traits > & **std::operator<<** (basic_ostream< _CharT, _Traits > &__out, thread::id __id)
- bool **std::operator<=** (thread::id __x, thread::id __y) noexcept
- bool **std::operator>** (thread::id __x, thread::id __y) noexcept
- bool **std::operator>=** (thread::id __x, thread::id __y) noexcept
- template<typename _Rep, typename _Period >
void **std::this_thread::sleep_for** (const chrono::duration< _Rep, _Period > &__rtime)
- template<typename _Clock, typename _Duration >
void **std::this_thread::sleep_until** (const chrono::time_point< _Clock, _Duration > &__atime)
- void **std::swap** (thread &__x, thread &__y) noexcept
- void **std::this_thread::yield** () noexcept

6.614.1 Detailed Description

This is a Standard C++ Library header.

6.615 throw_allocator.h File Reference

Classes

- struct [__gnu_cxx::annotate_base](#)
- struct [__gnu_cxx::condition_base](#)
- struct [__gnu_cxx::forced_error](#)
- struct [__gnu_cxx::limit_condition](#)
- struct [__gnu_cxx::limit_condition::always_adjustor](#)
- struct [__gnu_cxx::limit_condition::limit_adjustor](#)
- struct [__gnu_cxx::limit_condition::never_adjustor](#)
- struct [__gnu_cxx::random_condition](#)
- struct [__gnu_cxx::random_condition::always_adjustor](#)
- struct [__gnu_cxx::random_condition::group_adjustor](#)
- struct [__gnu_cxx::random_condition::never_adjustor](#)
- class [__gnu_cxx::throw_allocator_base< _Tp, _Cond >](#)
- struct [__gnu_cxx::throw_allocator_limit< _Tp >](#)
- struct [__gnu_cxx::throw_allocator_random< _Tp >](#)
- struct [__gnu_cxx::throw_value_base< _Cond >](#)
- struct [__gnu_cxx::throw_value_limit](#)
- struct [__gnu_cxx::throw_value_random](#)
- struct [std::hash< __gnu_cxx::throw_value_limit >](#)
- struct [std::hash< __gnu_cxx::throw_value_random >](#)

Namespaces

- [__gnu_cxx](#)
- [std](#)

Functions

- void **__gnu_cxx::__throw_forced_error** ()
- template<typename _Tp, typename _Cond >
bool **__gnu_cxx::operator!=** (const throw_allocator_base< _Tp, _Cond > &, const throw_allocator_base< _Tp, _Cond > &)
- template<typename _Cond >
throw_value_base< _Cond > **__gnu_cxx::operator*** (const throw_value_base< _Cond > &__a, const throw_value_base< _Cond > &__b)
- template<typename _Cond >
throw_value_base< _Cond > **__gnu_cxx::operator+** (const throw_value_base< _Cond > &__a, const throw_value_base< _Cond > &__b)
- template<typename _Cond >
throw_value_base< _Cond > **__gnu_cxx::operator-** (const throw_value_base< _Cond > &__a, const throw_value_base< _Cond > &__b)
- template<typename _Cond >
bool **__gnu_cxx::operator<** (const throw_value_base< _Cond > &__a, const throw_value_base< _Cond > &__b)
- [std::ostream](#) & **__gnu_cxx::operator<<** ([std::ostream](#) &os, const annotate_base &__b)
- template<typename _Cond >
bool **__gnu_cxx::operator==** (const throw_value_base< _Cond > &__a, const throw_value_base< _Cond > &__b)
- template<typename _Tp, typename _Cond >
bool **__gnu_cxx::operator==** (const throw_allocator_base< _Tp, _Cond > &, const throw_allocator_base< _Tp, _Cond > &)
- template<typename _Cond >
void **__gnu_cxx::swap** (throw_value_base< _Cond > &__a, throw_value_base< _Cond > &__b)

6.615.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

Contains two exception-generating types (throw_value, throw_allocator) intended to be used as value and allocator types while testing exception safety in templated containers and algorithms. The allocator has additional log and debug features. The exception generated is of type forced_exception_error.

6.616 time_members.h File Reference

Namespaces

- [std](#)

6.616.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<locale>`.

6.617 `trace_fn_imps.hpp` File Reference

6.617.1 Detailed Description

Contains an implementation class for a `binary_heap`.

6.618 `trace_fn_imps.hpp` File Reference

6.618.1 Detailed Description

Contains implementations of `cc_ht_map_`'s trace-mode functions.

6.619 `trace_fn_imps.hpp` File Reference

6.619.1 Detailed Description

Contains implementations of `gp_ht_map_`'s trace-mode functions.

6.620 `trace_fn_imps.hpp` File Reference

6.620.1 Detailed Description

Contains an implementation class for `left_child_next_sibling_heap_`.

6.621 `trace_fn_imps.hpp` File Reference

6.621.1 Detailed Description

Contains implementations of `lu_map_`.

6.622 `trace_fn_imps.hpp` File Reference

6.622.1 Detailed Description

Contains an implementation class for `pat_trie_`.

6.623 `trace_fn_imps.hpp` File Reference

6.623.1 Detailed Description

Contains an implementation for `rc_binomial_heap_`.

6.624 `trace_fn_imps.hpp` File Reference

6.624.1 Detailed Description

Contains an implementation class for `left_child_next_sibling_heap_`.

6.625 `traits.hpp` File Reference

Classes

- struct [__gnu_pbds::detail::bin_search_tree_traits< Key, Mapped, Cmp_Fn, Node_Update, Node, _Alloc >](#)
- struct [__gnu_pbds::detail::bin_search_tree_traits< Key, null_type, Cmp_Fn, Node_Update, Node, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

6.625.1 Detailed Description

Contains an implementation for `bin_search_tree_`.

6.626 `traits.hpp` File Reference

Classes

- struct [__gnu_pbds::detail::tree_traits< Key, Data, Cmp_Fn, Node_Update, Tag, _Alloc >](#)
- struct [__gnu_pbds::detail::trie_traits< Key, Data, _ATraits, Node_Update, Tag, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_DEBUG_VERIFY(_Cond)`

6.626.1 Detailed Description

Contains an implementation class for tree-like classes.

6.627 traits.hpp File Reference

Classes

- struct [__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >](#)
- struct [__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

6.627.1 Detailed Description

Contains an implementation class for ov_tree_.

6.628 traits.hpp File Reference

Classes

- struct [__gnu_pbds::detail::trie_traits< Key, Mapped, _ATraits, Node_Update, pat_trie_tag, _Alloc >](#)
- struct [__gnu_pbds::detail::trie_traits< Key, null_type, _ATraits, Node_Update, pat_trie_tag, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

6.628.1 Detailed Description

Contains an implementation class for pat_trie_.

6.629 traits.hpp File Reference

Classes

- struct [__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >](#)
- struct [__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

6.629.1 Detailed Description

Contains an implementation for `rb_tree_`.

6.630 traits.hpp File Reference

Classes

- struct [__gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >](#)
- struct [__gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

6.630.1 Detailed Description

Contains an implementation for `splay_tree_`.

6.631 tree_policy.hpp File Reference

Classes

- class [__gnu_pbds::tree_order_statistics_node_update< Node_Cltr, Node_Itr, Cmp_Fn, _Alloc >](#)

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_BRANCH_POLICY_BASE`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`

6.631.1 Detailed Description

Contains tree-related policies.

6.632 tree_trace_base.hpp File Reference

6.632.1 Detailed Description

Contains tree-related policies.

6.633 trie_policy.hpp File Reference

Classes

- class [__gnu_pbds::trie_order_statistics_node_update](#)< Node_Cltr, Node_Itr, _ATraits, _Alloc >
- class [__gnu_pbds::trie_prefix_search_node_update](#)< Node_Cltr, Node_Itr, _ATraits, _Alloc >
- struct [__gnu_pbds::trie_string_access_traits](#)< String, Min_E_Val, Max_E_Val, Reverse, _Alloc >

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_CLASS_T_DEC`
- `#define PB_DS_TRIE_POLICY_BASE`

6.633.1 Detailed Description

Contains trie-related policies.

6.634 trie_policy_base.hpp File Reference

Classes

- class [__gnu_pbds::detail::trie_policy_base](#)< Node_Cltr, Node_Itr, _ATraits, _Alloc >

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_CLASS_C_DEC`
- `#define PB_DS_CLASS_T_DEC`

6.634.1 Detailed Description

Contains an implementation of `trie_policy_base`.

6.635 `trie_string_access_traits_imp.hpp` File Reference

6.635.1 Detailed Description

Contains a policy for extracting character positions from a string for a vector-based PATRICIA tree

6.636 `tuple` File Reference

Classes

- struct `std::_Tuple_impl<_Idx, _Elements>`
- struct `std::_Tuple_impl<_Idx, _Head, _Tail...>`
- class `std::tuple<_Elements>`
- class `std::tuple<_Elements>`
- class `std::tuple<_T1, _T2>`
- struct `std::tuple_element<0, tuple<_Head, _Tail...>>`
- struct `std::tuple_element<__i, tuple<_Head, _Tail...>>`
- struct `std::tuple_size<tuple<_Elements...>>`
- struct `std::uses_allocator<tuple<_Types...>, _Alloc>`

Namespaces

- `std`

Macros

- `#define __cpp_lib_tuples_by_type`
- `#define _GLIBCXX_TUPLE`

Typedefs

- template<typename `_Tp`>
using `std::__empty_not_final` = typename conditional<__is_final(_Tp), false_type, __is_empty_non_tuple<↔
_Tp>>::type

Functions

- `template<std::size_t __i, typename _Head, typename... _Tail>`
`constexpr _Head & std::__get_helper (_Tuple_impl< __i, _Head, _Tail... > &__t) noexcept`
- `template<std::size_t __i, typename _Head, typename... _Tail>`
`constexpr const _Head & std::__get_helper (const _Tuple_impl< __i, _Head, _Tail... > &__t) noexcept`
- `template<typename _Head, size_t __i, typename... _Tail>`
`constexpr _Head & std::__get_helper2 (_Tuple_impl< __i, _Head, _Tail... > &__t) noexcept`
- `template<typename _Head, size_t __i, typename... _Tail>`
`constexpr const _Head & std::__get_helper2 (const _Tuple_impl< __i, _Head, _Tail... > &__t) noexcept`
- `template<typename... _Elements>`
`constexpr tuple< _Elements &&... > std::forward_as_tuple (_Elements &&...__args) noexcept`
- `template<std::size_t __i, typename... _Elements>`
`constexpr __tuple_element_t< __i, tuple< _Elements... > > & std::get (tuple< _Elements... > &__t) noexcept`
- `template<std::size_t __i, typename... _Elements>`
`constexpr const __tuple_element_t< __i, tuple< _Elements... > > & std::get (const tuple< _Elements... > &__t) noexcept`
- `template<std::size_t __i, typename... _Elements>`
`constexpr __tuple_element_t< __i, tuple< _Elements... > > && std::get (tuple< _Elements... > &&__t) noexcept`
- `template<typename _Tp, typename... _Types>`
`constexpr _Tp & std::get (tuple< _Types... > &__t) noexcept`
- `template<typename _Tp, typename... _Types>`
`constexpr _Tp && std::get (tuple< _Types... > &&__t) noexcept`
- `template<typename _Tp, typename... _Types>`
`constexpr const _Tp & std::get (const tuple< _Types... > &__t) noexcept`
- `template<typename... _Elements>`
`constexpr tuple< typename __decay_and_strip< _Elements >::__type... > std::make_tuple (_Elements &&...__args)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool std::operator!= (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool std::operator< (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool std::operator<= (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool std::operator== (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool std::operator> (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename... _TElements, typename... _UElements>`
`constexpr bool std::operator>= (const tuple< _TElements... > &__t, const tuple< _UElements... > &__u)`
- `template<typename... _Elements>`
`void std::swap (tuple< _Elements... > &__x, tuple< _Elements... > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename... _Elements>`
`constexpr tuple< _Elements &&... > std::tie (_Elements &&...__args) noexcept`
- `template<typename... _Tpls, typename = typename enable_if<__and<__is_tuple_like<_Tpls>...>::value>::type>`
`constexpr auto std::tuple_cat (_Tpls &&...__tpls) -> typename __tuple_cat_result< _Tpls... >::__type`

Variables

- `const _Swallow_assign std::ignore`

6.636.1 Detailed Description

This is a Standard C++ Library header.

6.637 tuple File Reference

Namespaces

- [std](#)

Macros

- `#define __cpp_lib_experimental_tuple`
- `#define _GLIBCXX_EXPERIMENTAL_TUPLE`

Functions

- `template<typename _Fn, typename _Tuple, std::size_t... _Idx>`
`decltype(auto) constexpr std::experimental::fundamentals_v1::__apply_impl (_Fn &&f, _Tuple &&t, std::index_sequence<_Idx...>)`
- `template<typename _Fn, typename _Tuple >`
`decltype(auto) constexpr std::experimental::fundamentals_v1::apply (_Fn &&f, _Tuple &&t)`

Variables

- `template<typename _Tp >`
`constexpr size_t std::experimental::fundamentals_v1::tuple_size_v`

6.637.1 Detailed Description

This is a TS C++ Library header.

6.638 `type_traits` File Reference

Classes

- struct [std::__is_nullptr_t< _Tp >](#)
- struct [std::integral_constant< _Tp, __v >](#)
- struct [std::is_abstract< _Tp >](#)
- struct [std::is_arithmetic< _Tp >](#)
- struct [std::is_array< typename >](#)
- struct [std::is_class< _Tp >](#)
- struct [std::is_compound< _Tp >](#)
- struct [std::is_const< typename >](#)
- struct [std::is_empty< _Tp >](#)
- struct [std::is_enum< _Tp >](#)
- struct [std::is_final< _Tp >](#)
- struct [std::is_floating_point< _Tp >](#)
- struct [std::is_function< typename >](#)
- struct [std::is_function< typename >](#)
- struct [std::is_fundamental< _Tp >](#)
- struct [std::is_integral< _Tp >](#)
- struct [std::is_literal_type< _Tp >](#)
- struct [std::is_lvalue_reference< typename >](#)
- struct [std::is_member_function_pointer< _Tp >](#)
- struct [std::is_member_object_pointer< _Tp >](#)
- struct [std::is_member_pointer< _Tp >](#)
- struct [std::is_member_pointer< _Tp >](#)
- struct [std::is_null_pointer< _Tp >](#)
- struct [std::is_object< _Tp >](#)
- struct [std::is_pod< _Tp >](#)
- struct [std::is_pointer< _Tp >](#)
- struct [std::is_polymorphic< _Tp >](#)
- struct [std::is_reference< _Tp >](#)
- struct [std::is_rvalue_reference< typename >](#)
- struct [std::is_scalar< _Tp >](#)
- struct [std::is_standard_layout< _Tp >](#)
- struct [std::is_trivial< _Tp >](#)
- struct [std::is_union< _Tp >](#)
- struct [std::is_void< _Tp >](#)
- struct [std::is_volatile< typename >](#)

Namespaces

- [std](#)

Macros

- `#define __cpp_lib_integral_constant_callable`
- `#define __cpp_lib_is_final`
- `#define __cpp_lib_is_null_pointer`
- `#define __cpp_lib_result_of_sfinae`
- `#define __cpp_lib_transformation_trait_aliases`
- `#define __cpp_lib_void_t`
- `#define _GLIBCXX_HAS_NESTED_TYPE(_NTYPE)`
- `#define _GLIBCXX_TYPE_TRAITS`

Typedefs

- `template<bool __v>`
`using std::__bool_constant = integral_constant< bool, __v >`
- `typedef integral_constant< bool, false > std::false_type`
- `typedef integral_constant< bool, true > std::true_type`

6.638.1 Detailed Description

This is a Standard C++ Library header.

6.639 `type_traits` File Reference

Classes

- `struct std::tr2::__reflection_typelist< _Elements >`
- `struct std::tr2::__reflection_typelist< _First, _Rest... >`
- `struct std::tr2::__reflection_typelist<>`
- `struct std::tr2::bases< _Tp >`
- `struct std::tr2::direct_bases< _Tp >`

Namespaces

- [std](#)
- [std::tr2](#)

Macros

- `#define _GLIBCXX_TR2_TYPE_TRAITS`

6.639.1 Detailed Description

This is a TR2 C++ Library header.

6.640 type_traits File Reference

Namespaces

- [std](#)

Macros

- `#define __cpp_lib_experimental_detect`
- `#define __cpp_lib_experimental_logical_traits`
- `#define __cpp_lib_experimental_type_trait_variable_templates`
- `#define _GLIBCXX_EXPERIMENTAL_TYPE_TRAITS`

Typedefs

- `template<typename _Default, template< typename... > class _Op, typename... _Args>`
using **`std::experimental::fundamentals_v2::detected_or`** = `std::__detected_or< _Default, _Op, _Args... >`
- `template<typename _Default, template< typename... > class _Op, typename... _Args>`
using **`std::experimental::fundamentals_v2::detected_or_t`** = `typename detected_or< _Default, _Op, _Args... >::type`
- `template<template< typename... > class _Op, typename... _Args>`
using **`std::experimental::fundamentals_v2::detected_t`** = `typename std::__detector< nonesuch, void, _Op, ↵
_Args... >::type`
- `template<template< typename... > class _Op, typename... _Args>`
using **`std::experimental::fundamentals_v2::is_detected`** = `typename std::__detector< nonesuch, void, _Op, ↵
_Args... >::value_t`
- `template<typename _To, template< typename... > class _Op, typename... _Args>`
using **`std::experimental::fundamentals_v2::is_detected_convertible`** = `is_convertible< detected_t< _Op, ↵
_Args... >, _To >`
- `template<typename Expected, template< typename... > class _Op, typename... _Args>`
using **`std::experimental::fundamentals_v2::is_detected_exact`** = `is_same< Expected, detected_t< _Op, ↵
Args... >>`
- `template<typename... >`
using **`std::experimental::fundamentals_v2::void_t`** = `void`

Variables

- `template<typename _Tp >`
`constexpr size_t std::experimental::fundamentals_v1::alignment_of_v`
- `template<typename... _Bn>`
`constexpr bool std::experimental::fundamentals_v2::conjunction_v`
- `template<typename... _Bn>`
`constexpr bool std::experimental::fundamentals_v2::disjunction_v`
- `template<typename _Tp, unsigned _Idx = 0>`
`constexpr size_t std::experimental::fundamentals_v1::extent_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::has_virtual_destructor_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_abstract_v`

- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_arithmetic_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_array_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v1::is_assignable_v`
- `template<typename _Base, typename _Derived >`
`constexpr bool std::experimental::fundamentals_v1::is_base_of_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_class_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_compound_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_const_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool std::experimental::fundamentals_v1::is_constructible_v`
- `template<typename _From, typename _To >`
`constexpr bool std::experimental::fundamentals_v1::is_convertible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_destructible_v`
- `template<typename _To, template< typename... > class _Op, typename... _Args>`
`constexpr bool std::experimental::fundamentals_v2::is_detected_convertible_v`
- `template<typename Expected, template< typename... > class _Op, typename... _Args>`
`constexpr bool std::experimental::fundamentals_v2::is_detected_exact_v`
- `template<template< typename... > class _Op, typename... _Args>`
`constexpr bool std::experimental::fundamentals_v2::is_detected_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_empty_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_enum_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_final_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_floating_point_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_function_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_fundamental_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_integral_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_literal_type_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_lvalue_reference_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_member_function_pointer_v`

- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_member_object_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_member_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_move_constructible_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v1::is_nothrow_assignable_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool std::experimental::fundamentals_v1::is_nothrow_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_nothrow_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_nothrow_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_nothrow_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_nothrow_destructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_nothrow_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_nothrow_move_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_null_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_object_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_pod_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_pointer_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_polymorphic_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_reference_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_rvalue_reference_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v1::is_same_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_scalar_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_signed_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_standard_layout_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_trivial_v`
- `template<typename _Tp, typename _Up >`
`constexpr bool std::experimental::fundamentals_v1::is_trivially_assignable_v`
- `template<typename _Tp, typename... _Args>`
`constexpr bool std::experimental::fundamentals_v1::is_trivially_constructible_v`

- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_trivially_copy_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_trivially_copy_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_trivially_copyable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_trivially_default_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_trivially_destructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_trivially_move_assignable_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_trivially_move_constructible_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_union_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_unsigned_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_void_v`
- `template<typename _Tp >`
`constexpr bool std::experimental::fundamentals_v1::is_volatile_v`
- `template<typename _Pp >`
`constexpr bool std::experimental::fundamentals_v2::negation_v`
- `template<typename _Tp >`
`constexpr size_t std::experimental::fundamentals_v1::rank_v`

6.640.1 Detailed Description

This is a TS C++ Library header.

6.641 `type_traits.h` File Reference

Namespaces

- [`__gnu_cxx`](#)

Functions

- `template<typename _Type >`
`bool __gnu_cxx::__is_null_pointer (_Type * __ptr)`
- `template<typename _Type >`
`bool __gnu_cxx::__is_null_pointer (_Type)`
- `bool __gnu_cxx::__is_null_pointer (std::nullptr_t)`

6.641.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.642 type_utils.hpp File Reference

Namespaces

- [__gnu_pbds](#)

Macros

- `#define PB_DS_STATIC_ASSERT(UNIQUE, E)`

Typedefs

- `typedef std::tr1::integral_constant< int, 0 > __gnu_pbds::detail::false_type`
- `typedef std::tr1::integral_constant< int, 1 > __gnu_pbds::detail::true_type`

6.642.1 Detailed Description

Contains utilities for handling types. All of these classes are based on Modern C++ by Andrei Alexandrescu.

6.643 typeindex File Reference

Classes

- struct [std::hash< _Tp >](#)
- struct [std::hash< type_index >](#)
- struct [std::type_index](#)

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_TYPEINDEX`

6.643.1 Detailed Description

This is a Standard C++ Library header.

6.644 typeinfo File Reference

Classes

- class [std::bad_cast](#)
- class [std::bad_typeid](#)
- class [std::type_info](#)

Namespaces

- [std](#)

Macros

- `#define __GXX_MERGED_TYPEINFO_NAMES`
- `#define __GXX_TYPEINFO_EQUALITY_INLINE`
- `#define _TYPEINFO`

6.644.1 Detailed Description

This is a Standard C++ Library header.

6.645 typelist.h File Reference

Namespaces

- [__gnu_cxx](#)
- [__gnu_cxx::typelist](#)

Macros

- `#define _GLIBCXX_TPELIST_CHAIN1(X0)`
- `#define _GLIBCXX_TPELIST_CHAIN10(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9)`
- `#define _GLIBCXX_TPELIST_CHAIN11(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10)`
- `#define _GLIBCXX_TPELIST_CHAIN12(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10, X11)`
- `#define _GLIBCXX_TPELIST_CHAIN13(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10, X11, X12)`
- `#define _GLIBCXX_TPELIST_CHAIN14(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10, X11, X12, X13)`
- `#define _GLIBCXX_TPELIST_CHAIN15(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10, X11, X12, X13, X14)`
- `#define _GLIBCXX_TPELIST_CHAIN16(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10, X11, X12, X13, X14, X15)`
- `#define _GLIBCXX_TPELIST_CHAIN17(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10, X11, X12, X13, X14, X15, X16)`
- `#define _GLIBCXX_TPELIST_CHAIN18(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10, X11, X12, X13, X14, X15, X16, X17)`

- `#define _GLIBCXX_TYPELIST_CHAIN19(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10, X11, X12, X13, X14, X15, X16, X17, X18)`
- `#define _GLIBCXX_TYPELIST_CHAIN2(X0, X1)`
- `#define _GLIBCXX_TYPELIST_CHAIN20(X0, X1, X2, X3, X4, X5, X6, X7, X8, X9, X10, X11, X12, X13, X14, X15, X16, X17, X18, X19)`
- `#define _GLIBCXX_TYPELIST_CHAIN3(X0, X1, X2)`
- `#define _GLIBCXX_TYPELIST_CHAIN4(X0, X1, X2, X3)`
- `#define _GLIBCXX_TYPELIST_CHAIN5(X0, X1, X2, X3, X4)`
- `#define _GLIBCXX_TYPELIST_CHAIN6(X0, X1, X2, X3, X4, X5)`
- `#define _GLIBCXX_TYPELIST_CHAIN7(X0, X1, X2, X3, X4, X5, X6)`
- `#define _GLIBCXX_TYPELIST_CHAIN8(X0, X1, X2, X3, X4, X5, X6, X7)`
- `#define _GLIBCXX_TYPELIST_CHAIN9(X0, X1, X2, X3, X4, X5, X6, X7, X8)`

Functions

- `template<typename Fn, typename Typelist >`
`void __gnu_cxx::typelist::apply (Fn &, Typelist)`
- `template<typename Gn, typename Typelist >`
`void __gnu_cxx::typelist::apply_generator (Gn &, Typelist)`
- `template<typename Gn, typename TypelistT, typename TypelistV >`
`void __gnu_cxx::typelist::apply_generator (Gn &, TypelistT, TypelistV)`
- `template<typename Fn, typename Typelist >`
`void __gnu_cxx::typelist::apply_generator (Fn &fn, Typelist)`
- `template<typename Fn, typename TypelistT, typename TypelistV >`
`void __gnu_cxx::typelist::apply_generator (Fn &fn, TypelistT, TypelistV)`

6.645.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

Contains `typelist_chain` definitions. Typelists are an idea by Andrei Alexandrescu.

6.646 types.h File Reference

Namespaces

- [__gnu_parallel](#)

Typedefs

- `typedef int64_t __gnu_parallel::CASable`
- `typedef uint64_t __gnu_parallel::SequenceIndex`
- `typedef uint16_t __gnu_parallel::ThreadIndex`

Enumerations

- enum `__gnu_parallel::AlgorithmStrategy` { **heuristic**, **force_sequential**, **force_parallel** }
- enum `__gnu_parallel::FindAlgorithm` { **GROWING_BLOCKS**, **CONSTANT_SIZE_BLOCKS**, **EQUAL_SPLIT** }
- enum `__gnu_parallel::MultiwayMergeAlgorithm` { **LOSER_TREE** }
- enum `__gnu_parallel::Parallelism` {
`__gnu_parallel::sequential`, `__gnu_parallel::parallel_unbalanced`, `__gnu_parallel::parallel_balanced`, `__gnu_parallel::parallel_omp_loop`,
`__gnu_parallel::parallel_omp_loop_static`, `__gnu_parallel::parallel_taskqueue` }
- enum `__gnu_parallel::PartialSumAlgorithm` { **RECURSIVE**, **LINEAR** }
- enum `__gnu_parallel::SortAlgorithm` { **MWMS**, **QS**, **QS_BALANCED** }
- enum `__gnu_parallel::SplittingAlgorithm` { **SAMPLING**, **EXACT** }

Variables

- static const int `__gnu_parallel::_CASable_bits`
- static const `_CASable` `__gnu_parallel::_CASable_mask`

6.646.1 Detailed Description

Basic types and typedefs. This file is a GNU parallel extension to the Standard C++ Library.

6.647 `types_traits.hpp` File Reference

Classes

- struct `__gnu_pbds::detail::no_throw_copies< Key, Mapped >`
- struct `__gnu_pbds::detail::no_throw_copies< Key, null_type >`
- struct `__gnu_pbds::detail::stored_data< _Tv, _Th >`
- struct `__gnu_pbds::detail::stored_data< _Tv, null_type >`
- struct `__gnu_pbds::detail::stored_hash< _Th >`
- struct `__gnu_pbds::detail::stored_value< _Tv >`
- struct `__gnu_pbds::detail::type_base< Key, Mapped, _Alloc, Store_Hash >`
- struct `__gnu_pbds::detail::type_base< Key, Mapped, _Alloc, false >`
- struct `__gnu_pbds::detail::type_base< Key, Mapped, _Alloc, true >`
- struct `__gnu_pbds::detail::type_base< Key, null_type, _Alloc, false >`
- struct `__gnu_pbds::detail::type_base< Key, null_type, _Alloc, true >`
- struct `__gnu_pbds::detail::type_dispatch< Key, Mapped, _Alloc, Store_Hash >`
- struct `__gnu_pbds::detail::types_traits< Key, Mapped, _Alloc, Store_Hash >`

Namespaces

- `__gnu_pbds`

6.647.1 Detailed Description

Contains a traits class of types used by containers.

6.648 `uniform_int_dist.h` File Reference

Classes

- class `std::uniform_int_distribution<_IntType>`
- struct `std::uniform_int_distribution<_IntType>::param_type`

Namespaces

- `std`
- `std::__detail`

Functions

- template<typename `_Tp`>
bool `std::__detail::__Power_of_2` (`_Tp __x`)

6.648.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<random>`.

6.649 `unique_copy.h` File Reference

Namespaces

- `__gnu_parallel`

Functions

- template<typename `_Iter`, class `_OutputIterator`, class `_BinaryPredicate`>
`_OutputIterator` `__gnu_parallel::__parallel_unique_copy` (`_Iter __first`, `_Iter __last`, `_OutputIterator __result`, `__gnu_parallel::__BinaryPredicate __binary_pred`)
- template<typename `_Iter`, class `_OutputIterator`>
`_OutputIterator` `__gnu_parallel::__parallel_unique_copy` (`_Iter __first`, `_Iter __last`, `_OutputIterator __result`)

6.649.1 Detailed Description

Parallel implementations of `std::unique_copy()`. This file is a GNU parallel extension to the Standard C++ Library.

6.650 `unique_ptr.h` File Reference

Classes

- struct `std::default_delete<_Tp>`
- struct `std::default_delete<_Tp[]>`
- struct `std::hash< unique_ptr<_Tp, _Dp>>`
- class `std::unique_ptr<_Tp, _Dp>`
- class `std::unique_ptr<_Tp[], _Dp>`

Namespaces

- `std`

Macros

- `#define __cpp_lib_make_unique`

Functions

- `template<typename _Tp, typename... _Args>`
`_MakeUniq<_Tp>::__single_object std::make_unique (_Args &&... __args)`
- `template<typename _Tp>`
`_MakeUniq<_Tp>::__array std::make_unique (size_t __num)`
- `template<typename _Tp, typename... _Args>`
`_MakeUniq<_Tp>::__invalid_type std::make_unique (_Args &&...)=delete`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep>`
`bool std::operator!= (const unique_ptr<_Tp, _Dp> &__x, const unique_ptr<_Up, _Ep> &__y)`
- `template<typename _Tp, typename _Dp>`
`bool std::operator!= (const unique_ptr<_Tp, _Dp> &__x, nullptr_t) noexcept`
- `template<typename _Tp, typename _Dp>`
`bool std::operator!= (nullptr_t, const unique_ptr<_Tp, _Dp> &__x) noexcept`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep>`
`bool std::operator< (const unique_ptr<_Tp, _Dp> &__x, const unique_ptr<_Up, _Ep> &__y)`
- `template<typename _Tp, typename _Dp>`
`bool std::operator< (const unique_ptr<_Tp, _Dp> &__x, nullptr_t)`
- `template<typename _Tp, typename _Dp>`
`bool std::operator< (nullptr_t, const unique_ptr<_Tp, _Dp> &__x)`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep>`
`bool std::operator<= (const unique_ptr<_Tp, _Dp> &__x, const unique_ptr<_Up, _Ep> &__y)`
- `template<typename _Tp, typename _Dp>`
`bool std::operator<= (const unique_ptr<_Tp, _Dp> &__x, nullptr_t)`
- `template<typename _Tp, typename _Dp>`
`bool std::operator<= (nullptr_t, const unique_ptr<_Tp, _Dp> &__x)`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep>`
`bool std::operator== (const unique_ptr<_Tp, _Dp> &__x, const unique_ptr<_Up, _Ep> &__y)`
- `template<typename _Tp, typename _Dp>`
`bool std::operator== (const unique_ptr<_Tp, _Dp> &__x, nullptr_t) noexcept`

- `template<typename _Tp, typename _Dp >`
`bool std::operator== (nullptr_t, const unique_ptr< _Tp, _Dp > &__x) noexcept`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`bool std::operator> (const unique_ptr< _Tp, _Dp > &__x, const unique_ptr< _Up, _Ep > &__y)`
- `template<typename _Tp, typename _Dp >`
`bool std::operator> (const unique_ptr< _Tp, _Dp > &__x, nullptr_t)`
- `template<typename _Tp, typename _Dp >`
`bool std::operator> (nullptr_t, const unique_ptr< _Tp, _Dp > &__x)`
- `template<typename _Tp, typename _Dp, typename _Up, typename _Ep >`
`bool std::operator>= (const unique_ptr< _Tp, _Dp > &__x, const unique_ptr< _Up, _Ep > &__y)`
- `template<typename _Tp, typename _Dp >`
`bool std::operator>= (const unique_ptr< _Tp, _Dp > &__x, nullptr_t)`
- `template<typename _Tp, typename _Dp >`
`bool std::operator>= (nullptr_t, const unique_ptr< _Tp, _Dp > &__x)`
- `template<typename _Tp, typename _Dp >`
`void std::swap (unique_ptr< _Tp, _Dp > &__x, unique_ptr< _Tp, _Dp > &__y) noexcept`

6.650.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<memory>`.

6.651 unordered_base.h File Reference

Namespaces

- [std](#)
- [std::__profile](#)

Functions

- `template<typename _UnorderedCont, typename _Value, bool _Cache_hash_code>`
`bool std::__profile::__are_equal (const _UnorderedCont &__uc, const __detail::__Hash_node< _Value, _Cache_hash_code > *__lhs, const __detail::__Hash_node< _Value, _Cache_hash_code > *__rhs)`
- `template<typename _UnorderedCont, typename _Value, bool _Cache_hash_code>`
`std::size_t std::__profile::__get_bucket_index (const _UnorderedCont &__uc, const __detail::__Hash_node< _Value, _Cache_hash_code > *__node)`

6.651.1 Detailed Description

This file is a GNU profile extension to the Standard C++ Library.

6.652 unordered_map File Reference

Macros

- `#define _GLIBCXX_UNORDERED_MAP`

6.652.1 Detailed Description

This is a Standard C++ Library header.

6.653 unordered_map File Reference

Classes

- class [std::__debug::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>](#)
- class [std::__debug::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>](#)

Namespaces

- [std](#)
- [std::__debug](#)

Macros

- `#define _GLIBCXX_DEBUG_UNORDERED_MAP`

Functions

- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc>
bool std::__debug::operator!= (const unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc> &__x, const unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc>
bool std::__debug::operator!= (const unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc> &__x, const unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc>
bool std::__debug::operator== (const unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc> &__x, const unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc>
bool std::__debug::operator== (const unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc> &__x, const unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc>
void std::__debug::swap (unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc> &__x, unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc> &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc>
void std::__debug::swap (unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc> &__x, unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc> &__y) noexcept(noexcept(__x.swap(__y)))`

6.653.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.654 unordered_map File Reference

Classes

- class [std::__profile::unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc>](#)
- class [std::__profile::unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc>](#)

Namespaces

- [std](#)
- [std::__profile](#)

Macros

- `#define _GLIBCXX_BASE`
- `#define _GLIBCXX_BASE`
- `#define _GLIBCXX_PROFILE_UNORDERED_MAP`
- `#define _GLIBCXX_STD_BASE`
- `#define _GLIBCXX_STD_BASE`

Functions

- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc>
bool std::__profile::operator!= (const unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc> &__x, const
unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc>
bool std::__profile::operator!= (const unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc> &__x, const
unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc>
bool std::__profile::operator== (const unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc> &__x, const
unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc>
bool std::__profile::operator== (const unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc> &__x, const
unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc> &__y)`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc>
void std::__profile::swap (unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc> &__x, unordered_map<_Key,
_Tp, _Hash, _Pred, _Alloc> &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Key, typename _Tp, typename _Hash, typename _Pred, typename _Alloc>
void std::__profile::swap (unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc> &__x, unordered_
multimap<_Key, _Tp, _Hash, _Pred, _Alloc> &__y) noexcept(noexcept(__x.swap(__y)))`

6.654.1 Detailed Description

This file is a GNU profile extension to the Standard C++ Library.

6.655 unordered_map File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_UNORDERED_MAP`

Typedefs

- `template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_to<_Key>>>
using std::experimental::fundamentals_v2::pmr::unordered_map = std::unordered_map<_Key, _Tp, _Hash,
_Pred, polymorphic_allocator< pair< const _Key, _Tp >>>>`
- `template<typename _Key , typename _Tp , typename _Hash = hash<_Key>, typename _Pred = equal_to<_Key>>>
using std::experimental::fundamentals_v2::pmr::unordered_multimap = std::unordered_multimap<_Key, ↵
_Tp, _Hash, _Pred, polymorphic_allocator< pair< const _Key, _Tp >>>>`

Functions

- `template<typename _Key , typename _Tp , typename _Hash , typename _CPred , typename _Alloc , typename _Predicate >
void std::experimental::fundamentals_v2::erase_if (unordered_map< _Key, _Tp, _Hash, _CPred, _Alloc >
&__cont, _Predicate __pred)`
- `template<typename _Key , typename _Tp , typename _Hash , typename _CPred , typename _Alloc , typename _Predicate >
void std::experimental::fundamentals_v2::erase_if (unordered_multimap< _Key, _Tp, _Hash, _CPred, _Alloc
> &__cont, _Predicate __pred)`

6.655.1 Detailed Description

This is a TS C++ Library header.

6.656 unordered_map.h File Reference

Classes

- class [std::unordered_map](#)<_Key, _Tp, _Hash, _Pred, _Alloc >
- class [std::unordered_multimap](#)<_Key, _Tp, _Hash, _Pred, _Alloc >

Namespaces

- [std](#)

Typedefs

- `template<typename _Key, typename _Tp, typename _Hash = hash<_Key>, typename _Pred = std::equal_to<_Key>, typename _Alloc = std::allocator<std::pair<const _Key, _Tp>>, typename _Tr = __umap_traits<__cache_default<_Key, _Hash>::value>>
using std::__umap_hashtable = _Hashtable<_Key, std::pair< const _Key, _Tp >, _Alloc, __detail::__Select1st, _Pred, _Hash, __detail::__Mod_range_hashing, __detail::__Default_ranged_hash, __detail::__Prime_rehash_policy, _Tr >`
- `template<bool _Cache>
using std::__umap_traits = __detail::__Hashtable_traits< _Cache, false, true >`
- `template<typename _Key, typename _Tp, typename _Hash = hash<_Key>, typename _Pred = std::equal_to<_Key>, typename _Alloc = std::allocator<std::pair<const _Key, _Tp>>, typename _Tr = __ummap_traits<__cache_default<_Key, _Hash>::value>>
using std::__ummap_hashtable = _Hashtable<_Key, std::pair< const _Key, _Tp >, _Alloc, __detail::__Select1st, _Pred, _Hash, __detail::__Mod_range_hashing, __detail::__Default_ranged_hash, __detail::__Prime_rehash_policy, _Tr >`
- `template<bool _Cache>
using std::__ummap_traits = __detail::__Hashtable_traits< _Cache, false, false >`

Functions

- `template<class _Key, class _Tp, class _Hash, class _Pred, class _Alloc >
bool std::operator!= (const unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc > &__x, const unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc > &__y)`
- `template<class _Key, class _Tp, class _Hash, class _Pred, class _Alloc >
bool std::operator!= (const unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc > &__x, const unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc > &__y)`
- `template<class _Key, class _Tp, class _Hash, class _Pred, class _Alloc >
bool std::operator== (const unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc > &__x, const unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc > &__y)`
- `template<class _Key, class _Tp, class _Hash, class _Pred, class _Alloc >
bool std::operator== (const unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc > &__x, const unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc > &__y)`
- `template<class _Key, class _Tp, class _Hash, class _Pred, class _Alloc >
void std::swap (unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc > &__x, unordered_map<_Key, _Tp, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<class _Key, class _Tp, class _Hash, class _Pred, class _Alloc >
void std::swap (unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc > &__x, unordered_multimap<_Key, _Tp, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`

6.656.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<unordered_map>`.

6.657 unordered_set File Reference

Macros

- `#define _GLIBCXX_UNORDERED_SET`

6.657.1 Detailed Description

This is a Standard C++ Library header.

6.658 unordered_set File Reference

Classes

- class [std::__debug::unordered_multiset< _Value, _Hash, _Pred, _Alloc >](#)
- class [std::__debug::unordered_set< _Value, _Hash, _Pred, _Alloc >](#)

Namespaces

- [std](#)
- [std::__debug](#)

Macros

- `#define _GLIBCXX_DEBUG_UNORDERED_SET`

Functions

- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc >`
`bool std::__debug::operator!= (const unordered_set< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_set< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc >`
`bool std::__debug::operator!= (const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc >`
`bool std::__debug::operator== (const unordered_set< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_set< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc >`
`bool std::__debug::operator== (const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc >`
`void std::__debug::swap (unordered_set< _Value, _Hash, _Pred, _Alloc > &__x, unordered_set< _Value, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc >`
`void std::__debug::swap (unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`

6.658.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.659 unordered_set File Reference

Classes

- class [std::__profile::unordered_multiset<_Value, _Hash, _Pred, _Alloc>](#)
- class [std::__profile::unordered_set<_Key, _Hash, _Pred, _Alloc>](#)

Namespaces

- [std](#)
- [std::__profile](#)

Macros

- `#define _GLIBCXX_BASE`
- `#define _GLIBCXX_BASE`
- `#define _GLIBCXX_PROFILE_UNORDERED_SET`
- `#define _GLIBCXX_STD_BASE`
- `#define _GLIBCXX_STD_BASE`

Functions

- `template<typename _Key, typename _Hash, typename _Pred, typename _Alloc>
bool std::__profile::operator!= (const unordered_set<_Key, _Hash, _Pred, _Alloc> &__x, const unordered_set<_Key, _Hash, _Pred, _Alloc> &__y)`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc>
bool std::__profile::operator!= (const unordered_multiset<_Value, _Hash, _Pred, _Alloc> &__x, const unordered_multiset<_Value, _Hash, _Pred, _Alloc> &__y)`
- `template<typename _Key, typename _Hash, typename _Pred, typename _Alloc>
bool std::__profile::operator== (const unordered_set<_Key, _Hash, _Pred, _Alloc> &__x, const unordered_set<_Key, _Hash, _Pred, _Alloc> &__y)`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc>
bool std::__profile::operator== (const unordered_multiset<_Value, _Hash, _Pred, _Alloc> &__x, const unordered_multiset<_Value, _Hash, _Pred, _Alloc> &__y)`
- `template<typename _Key, typename _Hash, typename _Pred, typename _Alloc>
void std::__profile::swap (unordered_set<_Key, _Hash, _Pred, _Alloc> &__x, unordered_set<_Key, _Hash, _Pred, _Alloc> &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<typename _Value, typename _Hash, typename _Pred, typename _Alloc>
void std::__profile::swap (unordered_multiset<_Value, _Hash, _Pred, _Alloc> &__x, unordered_multiset<_Value, _Hash, _Pred, _Alloc> &__y) noexcept(noexcept(__x.swap(__y)))`

6.659.1 Detailed Description

This file is a GNU profile extension to the Standard C++ Library.

6.660 unordered_set File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_UNORDERED_SET`

Typedefs

- `template<typename _Key, typename _Hash = hash<_Key>, typename _Pred = equal_to<_Key>>>`
using **std::experimental::fundamentals_v2::pmr::unordered_multiset** = [std::unordered_multiset](#)< _Key, _Hash, _Pred, polymorphic_allocator< _Key >>
- `template<typename _Key, typename _Hash = hash<_Key>, typename _Pred = equal_to<_Key>>>`
using **std::experimental::fundamentals_v2::pmr::unordered_set** = [std::unordered_set](#)< _Key, _Hash, _Pred, polymorphic_allocator< _Key >>

Functions

- `template<typename _Key, typename _Hash, typename _CPred, typename _Alloc, typename _Predicate >`
void **std::experimental::fundamentals_v2::erase_if** (unordered_set< _Key, _Hash, _CPred, _Alloc > &__cont, _Predicate __pred)
- `template<typename _Key, typename _Hash, typename _CPred, typename _Alloc, typename _Predicate >`
void **std::experimental::fundamentals_v2::erase_if** (unordered_multiset< _Key, _Hash, _CPred, _Alloc > &__cont, _Predicate __pred)

6.660.1 Detailed Description

This is a TS C++ Library header.

6.661 unordered_set.h File Reference

Classes

- class [std::unordered_multiset](#)< _Value, _Hash, _Pred, _Alloc >
- class [std::unordered_set](#)< _Value, _Hash, _Pred, _Alloc >

Namespaces

- [std](#)

Typedefs

- `template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = std::equal_to<_Value>, typename _Alloc = std::allocator<_Value>, typename _Tr = __umset_traits<__cache_default<_Value, _Hash>::value>>
using std::__umset_hashtable = _Hashtable< _Value, _Value, _Alloc, __detail:: Identity, _Pred, _Hash, __detail:: Mod_range_hashing, __detail:: Default_ranged_hash, __detail:: Prime_rehash_policy, _Tr >`
- `template<bool _Cache>
using std::__umset_traits = __detail:: _Hashtable_traits< _Cache, true, false >`
- `template<typename _Value , typename _Hash = hash<_Value>, typename _Pred = std::equal_to<_Value>, typename _Alloc = std::allocator<_Value>, typename _Tr = __uset_traits<__cache_default<_Value, _Hash>::value>>
using std::__uset_hashtable = _Hashtable< _Value, _Value, _Alloc, __detail:: Identity, _Pred, _Hash, __detail:: Mod_range_hashing, __detail:: Default_ranged_hash, __detail:: Prime_rehash_policy, _Tr >`
- `template<bool _Cache>
using std::__uset_traits = __detail:: _Hashtable_traits< _Cache, true, true >`

Functions

- `template<class _Value , class _Hash , class _Pred , class _Alloc >
bool std::operator!= (const unordered_set< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_set< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<class _Value , class _Hash , class _Pred , class _Alloc >
bool std::operator!= (const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<class _Value , class _Hash , class _Pred , class _Alloc >
bool std::operator== (const unordered_set< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_set< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<class _Value , class _Hash , class _Pred , class _Alloc >
bool std::operator== (const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, const unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__y)`
- `template<class _Value , class _Hash , class _Pred , class _Alloc >
void std::swap (unordered_set< _Value, _Hash, _Pred, _Alloc > &__x, unordered_set< _Value, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`
- `template<class _Value , class _Hash , class _Pred , class _Alloc >
void std::swap (unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__x, unordered_multiset< _Value, _Hash, _Pred, _Alloc > &__y) noexcept(noexcept(__x.swap(__y)))`

6.661.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<unordered_set>`.

6.662 update_fn_imps.hpp File Reference

6.662.1 Detailed Description

Contains an implementation class for `pat_trie_`.

6.663 utility File Reference

Classes

- struct [std::__is_tuple_like_impl< std::pair< _T1, _T2 > >](#)
- struct [std::integer_sequence< _Tp, _Idx >](#)
- struct [std::tuple_element< _Int, _Tp >](#)
- struct [std::tuple_element< 0, std::pair< _Tp1, _Tp2 > >](#)
- struct [std::tuple_element< 1, std::pair< _Tp1, _Tp2 > >](#)
- struct [std::tuple_size< _Tp >](#)
- struct [std::tuple_size< std::pair< _Tp1, _Tp2 > >](#)

Namespaces

- [std](#)

Macros

- `#define __cpp_lib_exchange_function`
- `#define __cpp_lib_integer_sequence`
- `#define __cpp_lib_tuple_element_t`
- `#define __cpp_lib_tuples_by_type`
- `#define _GLIBCXX_UTILITY`

Typedefs

- `template<std::size_t __i, typename _Tp >`
using [std::__tuple_element_t](#) = typename tuple_element< __i, _Tp >::type
- `template<size_t... _Idx>`
using [std::index_sequence](#) = integer_sequence< size_t, _Idx... >
- `template<typename... _Types>`
using [std::index_sequence_for](#) = make_index_sequence< sizeof...(_Types)>
- `template<size_t _Num>`
using [std::make_index_sequence](#) = make_integer_sequence< size_t, _Num >
- `template<typename _Tp, _Tp _Num>`
using [std::make_integer_sequence](#) = typename _Make_integer_sequence< _Tp, _Num >::__type
- `template<std::size_t __i, typename _Tp >`
using [std::tuple_element_t](#) = typename tuple_element< __i, _Tp >::type

Functions

- `template<typename _Tp, typename _Up = _Tp>
_Tp std::exchange (_Tp &__obj, _Up &&__new_val)`
- `template<std::size_t _Int, class _Tp1, class _Tp2 >
constexpr tuple_element< _Int, std::pair< _Tp1, _Tp2 > >::type & std::get (std::pair< _Tp1, _Tp2 > &__in)
noexcept`
- `template<std::size_t _Int, class _Tp1, class _Tp2 >
constexpr tuple_element< _Int, std::pair< _Tp1, _Tp2 > >::type && std::get (std::pair< _Tp1, _Tp2 > &&__in)
noexcept`
- `template<std::size_t _Int, class _Tp1, class _Tp2 >
constexpr const tuple_element< _Int, std::pair< _Tp1, _Tp2 > >::type & std::get (const std::pair< _Tp1, _Tp2
> &__in) noexcept`
- `template<typename _Tp, typename _Up >
constexpr _Tp & std::get (pair< _Tp, _Up > &__p) noexcept`
- `template<typename _Tp, typename _Up >
constexpr const _Tp & std::get (const pair< _Tp, _Up > &__p) noexcept`
- `template<typename _Tp, typename _Up >
constexpr _Tp && std::get (pair< _Tp, _Up > &&__p) noexcept`
- `template<typename _Tp, typename _Up >
constexpr _Tp & std::get (pair< _Up, _Tp > &__p) noexcept`
- `template<typename _Tp, typename _Up >
constexpr const _Tp & std::get (const pair< _Up, _Tp > &__p) noexcept`
- `template<typename _Tp, typename _Up >
constexpr _Tp && std::get (pair< _Up, _Tp > &&__p) noexcept`

6.663.1 Detailed Description

This is a Standard C++ Library header.

6.664 utility File Reference

Namespaces

- [std](#)

Macros

- `#define _GLIBCXX_EXPERIMENTAL_UTILITY`

Typedefs

- using `std::experimental::fundamentals_v2::erased_type = std::__erased_type`

6.664.1 Detailed Description

This is a TS C++ Library header.

6.665 valarray File Reference

Classes

- class [std::gslice_array< _Tp >](#)
- class [std::indirect_array< _Tp >](#)
- class [std::mask_array< _Tp >](#)
- class [std::slice_array< _Tp >](#)
- class [std::valarray< _Tp >](#)
- class [std::valarray< _Tp >](#)

Namespaces

- [std](#)

Macros

- `#define _DEFINE_BINARY_OPERATOR(_Op, _Name)`
- `#define _DEFINE_VALARRAY_AUGMENTED_ASSIGNMENT(_Op, _Name)`
- `#define _DEFINE_VALARRAY_EXPR_AUGMENTED_ASSIGNMENT(_Op, _Name)`
- `#define _DEFINE_VALARRAY_UNARY_OPERATOR(_Op, _Name)`
- `#define _GLIBCXX_VALARRAY`

Functions

- `template<class _Tp >`
`_Tp * std::begin (valarray< _Tp > &__va)`
- `template<class _Tp >`
`const _Tp * std::begin (const valarray< _Tp > &__va)`
- `template<class _Tp >`
`_Tp * std::end (valarray< _Tp > &__va)`
- `template<class _Tp >`
`const _Tp * std::end (const valarray< _Tp > &__va)`
- `template<typename _Tp >`
`_Expr< _BinClos< __not_equal_to, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __not_equal_to, _Tp >::result_type > std::operator!= (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __not_equal_to, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __not_equal_to, _Tp >::result_type > std::operator!= (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __not_equal_to, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __not_equal_to, _Tp >::result_type > std::operator!= (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __modulus, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __modulus, _Tp >::result_type > std::operator% (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __modulus, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __modulus, _Tp >::result_type > std::operator% (const valarray< _Tp > &__v, const _Tp &__t)`

- `template<typename _Tp >`
`_Expr< _BinClos< __modulus, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __modulus, _Tp >::result_type > std::operator% (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __bitwise_and, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __bitwise_and, _Tp >::result_type > std::operator& (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __bitwise_and, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __bitwise_and, _Tp >::result_type > std::operator& (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __bitwise_and, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __bitwise_and, _Tp >::result_type > std::operator& (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __logical_and, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __logical_and, _Tp >::result_type > std::operator&& (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __logical_and, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __logical_and, _Tp >::result_type > std::operator&& (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __logical_and, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __logical_and, _Tp >::result_type > std::operator&& (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __multiplies, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __multiplies, _Tp >::result_type > std::operator* (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __multiplies, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __multiplies, _Tp >::result_type > std::operator* (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __multiplies, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __multiplies, _Tp >::result_type > std::operator* (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __plus, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __plus, _Tp >::result_type > std::operator+ (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __plus, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __plus, _Tp >::result_type > std::operator+ (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __plus, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __plus, _Tp >::result_type > std::operator+ (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __minus, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __minus, _Tp >::result_type > std::operator- (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __minus, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __minus, _Tp >::result_type > std::operator- (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __minus, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __minus, _Tp >::result_type > std::operator- (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __divides, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __divides, _Tp >::result_type > std::operator/ (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __divides, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __divides, _Tp >::result_type > std::operator/ (const valarray< _Tp > &__v, const _Tp &__t)`

- `template<typename _Tp >`
`_Expr< _BinClos< __divides, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __divides, _Tp >::result_`
`type > std::operator/ (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __less, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __less, _Tp >::result_type >`
`std::operator< (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __less, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __less, _Tp >::result_type >`
`std::operator< (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __less, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __less, _Tp >::result_type >`
`std::operator< (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __shift_left, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __shift_left, _Tp >::result_`
`type > std::operator<< (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __shift_left, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __shift_left, _Tp >::result_`
`type > std::operator<< (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __shift_left, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __shift_left, _Tp >::result_`
`type > std::operator<< (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __less_equal, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __less_equal, _Tp >::`
`result_type > std::operator<= (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __less_equal, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __less_equal, _Tp >::`
`result_type > std::operator<= (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __less_equal, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __less_equal, _Tp >::`
`result_type > std::operator<= (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __equal_to, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __equal_to, _Tp >::result_`
`type > std::operator== (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __equal_to, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __equal_to, _Tp >::`
`result_type > std::operator== (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __equal_to, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __equal_to, _Tp >::`
`result_type > std::operator== (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __greater, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __greater, _Tp >::result_`
`type > std::operator> (const valarray< _Tp > &__v, const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< __greater, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __greater, _Tp >::result_type`
`> std::operator> (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __greater, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __greater, _Tp >::result_`
`type > std::operator> (const _Tp &__t, const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _BinClos< __greater_equal, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __greater_equal, _Tp`
`>::result_type > std::operator>= (const valarray< _Tp > &__v, const valarray< _Tp > &__w)`
- `template<typename _Tp >`
`_Expr< _BinClos< __greater_equal, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __greater_equal, _Tp`
`>::result_type > std::operator>= (const _Tp &__t, const valarray< _Tp > &__v)`

- `template<typename _Tp >`
`_Expr< _BinClos< __greater_equal, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __greater_equal, _Tp >::result_type > std::operator>=` (const valarray< _Tp > &__v, const _Tp &__t)
- `template<typename _Tp >`
`_Expr< _BinClos< __shift_right, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __shift_right, _Tp >::result_type > std::operator>>` (const _Tp &__t, const valarray< _Tp > &__v)
- `template<typename _Tp >`
`_Expr< _BinClos< __shift_right, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __shift_right, _Tp >::result_type > std::operator>>` (const valarray< _Tp > &__v, const _Tp &__t)
- `template<typename _Tp >`
`_Expr< _BinClos< __shift_right, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __shift_right, _Tp >::result_type > std::operator>>` (const valarray< _Tp > &__v, const valarray< _Tp > &__w)
- `template<typename _Tp >`
`_Expr< _BinClos< __bitwise_xor, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __bitwise_xor, _Tp >::result_type > std::operator^` (const valarray< _Tp > &__v, const _Tp &__t)
- `template<typename _Tp >`
`_Expr< _BinClos< __bitwise_xor, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __bitwise_xor, _Tp >::result_type > std::operator^` (const _Tp &__t, const valarray< _Tp > &__v)
- `template<typename _Tp >`
`_Expr< _BinClos< __bitwise_xor, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __bitwise_xor, _Tp >::result_type > std::operator^` (const valarray< _Tp > &__v, const valarray< _Tp > &__w)
- `template<typename _Tp >`
`_Expr< _BinClos< __bitwise_or, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __bitwise_or, _Tp >::result_type > std::operator|` (const valarray< _Tp > &__v, const valarray< _Tp > &__w)
- `template<typename _Tp >`
`_Expr< _BinClos< __bitwise_or, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __bitwise_or, _Tp >::result_type > std::operator|` (const valarray< _Tp > &__v, const _Tp &__t)
- `template<typename _Tp >`
`_Expr< _BinClos< __bitwise_or, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __bitwise_or, _Tp >::result_type > std::operator|` (const _Tp &__t, const valarray< _Tp > &__v)
- `template<typename _Tp >`
`_Expr< _BinClos< __logical_or, _ValArray, _ValArray, _Tp, _Tp >, typename __fun< __logical_or, _Tp >::result_type > std::operator||` (const valarray< _Tp > &__v, const valarray< _Tp > &__w)
- `template<typename _Tp >`
`_Expr< _BinClos< __logical_or, _ValArray, _Constant, _Tp, _Tp >, typename __fun< __logical_or, _Tp >::result_type > std::operator||` (const valarray< _Tp > &__v, const _Tp &__t)
- `template<typename _Tp >`
`_Expr< _BinClos< __logical_or, _Constant, _ValArray, _Tp, _Tp >, typename __fun< __logical_or, _Tp >::result_type > std::operator||` (const _Tp &__t, const valarray< _Tp > &__v)

6.665.1 Detailed Description

This is a Standard C++ Library header.

6.666 valarray_after.h File Reference

Namespaces

- [std](#)

Macros

- `#define _DEFINE_EXPR_BINARY_FUNCTION(_Fun, _UFun)`
- `#define _DEFINE_EXPR_BINARY_OPERATOR(_Op, _Name)`
- `#define _DEFINE_EXPR_UNARY_FUNCTION(_Name, _UName)`
- `#define _DEFINE_EXPR_UNARY_OPERATOR(_Op, _Name)`

Functions

- `template<class _Dom >`
`_Expr< _UnClos< _Abs, _Expr, _Dom >, typename _Dom::value_type > std::abs (const _Expr< _Dom, type-`
`name _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Abs, _ValArray, _Tp >, _Tp > std::abs (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< _Acos, _Expr, _Dom >, typename _Dom::value_type > std::acos (const _Expr< _Dom,`
`typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Acos, _ValArray, _Tp >, _Tp > std::acos (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< _Asin, _Expr, _Dom >, typename _Dom::value_type > std::asin (const _Expr< _Dom, type-`
`name _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Asin, _ValArray, _Tp >, _Tp > std::asin (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< _Atan, _Expr, _Dom >, typename _Dom::value_type > std::atan (const _Expr< _Dom,`
`typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Atan, _ValArray, _Tp >, _Tp > std::atan (const valarray< _Tp > &__v)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< _Atan2, _Expr, _Expr, _Dom1, _Dom2 >, typename _Dom1::value_type > std::atan2 (const`
`_Expr< _Dom1, typename _Dom1::value_type > &__e1, const _Expr< _Dom2, typename _Dom2::value_type`
`> &__e2)`
- `template<class _Dom >`
`_Expr< _BinClos< _Atan2, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename _Dom::value_↵`
`_type > std::atan2 (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _↵`
`Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< _Atan2, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_↵`
`type > std::atan2 (const valarray< typename _Dom::valarray > &__v, const _Expr< _Dom, typename _Dom_↵`
`::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< _Atan2, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename _Dom::value_↵`
`_type > std::atan2 (const _Expr< _Dom, typename _Dom::value_type > &__e, const typename _Dom::value_↵`
`_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< _Atan2, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_↵`
`_type > std::atan2 (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type`
`> &__e)`
- `template<typename _Tp >`
`_Expr< _BinClos< _Atan2, _ValArray, _ValArray, _Tp, _Tp >, _Tp > std::atan2 (const valarray< _Tp > &__v,`
`const valarray< _Tp > &__w)`

- `template<typename _Tp >`
`_Expr< _BinClos< _Atan2, _ValArray, _Constant, _Tp, _Tp >, _Tp > std::atan2 (const valarray< _Tp > &__v,`
`const _Tp &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< _Atan2, _Constant, _ValArray, _Tp, _Tp >, _Tp > std::atan2 (const _Tp &__t, const`
`valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< _Cos, _Expr, _Dom >, typename _Dom::value_type > std::cos (const _Expr< _Dom, type-`
`name _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Cos, _ValArray, _Tp >, _Tp > std::cos (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< _Cosh, _Expr, _Dom >, typename _Dom::value_type > std::cosh (const _Expr< _Dom,`
`typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Cosh, _ValArray, _Tp >, _Tp > std::cosh (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< _Exp, _Expr, _Dom >, typename _Dom::value_type > std::exp (const _Expr< _Dom, type-`
`name _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Exp, _ValArray, _Tp >, _Tp > std::exp (const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Log, _ValArray, _Tp >, _Tp > std::log (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< _Log, _Expr, _Dom >, typename _Dom::value_type > std::log (const _Expr< _Dom, type-`
`name _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _UnClos< _Log10, _Expr, _Dom >, typename _Dom::value_type > std::log10 (const _Expr< _Dom,`
`typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Log10, _ValArray, _Tp >, _Tp > std::log10 (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __not_equal_to, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__not_equal_to, typename _Dom::value_type >::result_type > std::operator!= (const typename _Dom::value_↵`
`type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __not_equal_to, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun<`
`__not_equal_to, typename _Dom::value_type >::result_type > std::operator!= (const _Expr< _Dom, typename`
`_Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __not_equal_to, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__not_equal_to, typename _Dom::value_type >::result_type > std::operator!= (const valarray< typename ↵`
`_Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __not_equal_to, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __not_equal_to, type-`
`name _Dom1::value_type >::result_type > std::operator!= (const _Expr< _Dom1, typename _Dom1::value_↵`
`type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __not_equal_to, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun<`
`__not_equal_to, typename _Dom::value_type >::result_type > std::operator!= (const _Expr< _Dom, typename`
`_Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __modulus, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun<`

```

__modulus, typename _Dom::value_type >::result_type > std::operator% (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)
• template<class _Dom1, class _Dom2 >
  _Expr< _BinClos< __modulus, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __modulus, typename _Dom1::value_type >::result_type > std::operator% (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)
• template<class _Dom >
  _Expr< _BinClos< __modulus, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __modulus, typename _Dom::value_type >::result_type > std::operator% (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)
• template<class _Dom >
  _Expr< _BinClos< __modulus, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< __modulus, typename _Dom::value_type >::result_type > std::operator% (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)
• template<class _Dom >
  _Expr< _BinClos< __modulus, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __modulus, typename _Dom::value_type >::result_type > std::operator% (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)
• template<class _Dom >
  _Expr< _BinClos< __bitwise_and, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< __bitwise_and, typename _Dom::value_type >::result_type > std::operator& (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)
• template<class _Dom >
  _Expr< _BinClos< __bitwise_and, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __bitwise_and, typename _Dom::value_type >::result_type > std::operator& (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)
• template<class _Dom >
  _Expr< _BinClos< __bitwise_and, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< __bitwise_and, typename _Dom::value_type >::result_type > std::operator& (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)
• template<class _Dom >
  _Expr< _BinClos< __bitwise_and, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __bitwise_and, typename _Dom::value_type >::result_type > std::operator& (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)
• template<class _Dom1, class _Dom2 >
  _Expr< _BinClos< __bitwise_and, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __bitwise_and, typename _Dom1::value_type >::result_type > std::operator& (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)
• template<class _Dom1, class _Dom2 >
  _Expr< _BinClos< __logical_and, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __logical_and, typename _Dom1::value_type >::result_type > std::operator&& (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)
• template<class _Dom >
  _Expr< _BinClos< __logical_and, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __logical_and, typename _Dom::value_type >::result_type > std::operator&& (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)
• template<class _Dom >
  _Expr< _BinClos< __logical_and, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< __logical_and, typename _Dom::value_type >::result_type > std::operator&& (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)
• template<class _Dom >
  _Expr< _BinClos< __logical_and, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __logical_and, typename _Dom::value_type >::result_type > std::operator&& (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)

```

- `template<class _Dom >`
`_Expr< _BinClos< __logical_and, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun<`
`__logical_and, typename _Dom::value_type >::result_type > std::operator&& (const _Expr< _Dom, typename`
`_Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __multiplies, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __multiplies, typename`
`_Dom1::value_type >::result_type > std::operator* (const _Expr< _Dom1, typename _Dom1::value_type >`
`&__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __multiplies, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun<`
`__multiplies, typename _Dom::value_type >::result_type > std::operator* (const _Expr< _Dom, typename _Dom`
`::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __multiplies, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__multiplies, typename _Dom::value_type >::result_type > std::operator* (const typename _Dom::value_type`
`&__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __multiplies, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun<`
`__multiplies, typename _Dom::value_type >::result_type > std::operator* (const _Expr< _Dom, typename _Dom`
`::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __multiplies, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__multiplies, typename _Dom::value_type >::result_type > std::operator* (const valarray< typename _Dom`
`::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< __plus, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< __plus,`
`typename _Dom::value_type >::result_type > std::operator+ (const _Expr< _Dom, typename _Dom::value`
`type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __plus, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< __plus,`
`typename _Dom::value_type >::result_type > std::operator+ (const _Expr< _Dom, typename _Dom::value`
`type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __plus, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __plus,`
`typename _Dom::value_type >::result_type > std::operator+ (const valarray< typename _Dom::value_type >`
`&__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __plus, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __plus, typename _Dom1`
`::value_type >::result_type > std::operator+ (const _Expr< _Dom1, typename _Dom1::value_type > &__v,`
`const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __plus, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __plus,`
`typename _Dom::value_type >::result_type > std::operator+ (const typename _Dom::value_type &__t, const`
`_Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __minus, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __`
`minus, typename _Dom::value_type >::result_type > std::operator- (const typename _Dom::value_type &__t,`
`const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __minus, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< __`
`minus, typename _Dom::value_type >::result_type > std::operator- (const _Expr< _Dom, typename _Dom`
`::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`

- `template<class _Dom >`
`_Expr< _BinClos< __minus, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __`
`minus, typename _Dom::value_type >::result_type > std::operator- (const valarray< typename _Dom::value`
`_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __minus, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __minus, typename _Dom1`
`::value_type >::result_type > std::operator- (const _Expr< _Dom1, typename _Dom1::value_type > &__`
`v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __minus, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< __`
`minus, typename _Dom::value_type >::result_type > std::operator- (const _Expr< _Dom, typename _Dom`
`::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __divides, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __divides, typename`
`_Dom1::value_type >::result_type > std::operator/ (const _Expr< _Dom1, typename _Dom1::value_type > &__`
`v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __divides, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< __`
`divides, typename _Dom::value_type >::result_type > std::operator/ (const _Expr< _Dom, typename _Dom`
`::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __divides, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __`
`divides, typename _Dom::value_type >::result_type > std::operator/ (const typename _Dom::value_type &__t,`
`const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __divides, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< __`
`divides, typename _Dom::value_type >::result_type > std::operator/ (const _Expr< _Dom, typename _Dom`
`::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __divides, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __`
`divides, typename _Dom::value_type >::result_type > std::operator/ (const valarray< typename _Dom::value`
`_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __less, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __less, typename _Dom1`
`::value_type >::result_type > std::operator< (const _Expr< _Dom1, typename _Dom1::value_type > &__v,`
`const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __less, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< __less,`
`typename _Dom::value_type >::result_type > std::operator< (const _Expr< _Dom, typename _Dom::value`
`_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __less, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __less,`
`typename _Dom::value_type >::result_type > std::operator< (const typename _Dom::value_type &__t, const`
`_Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __less, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< __less,`
`typename _Dom::value_type >::result_type > std::operator< (const _Expr< _Dom, typename _Dom::value`
`_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __less, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __less,`
`typename _Dom::value_type >::result_type > std::operator< (const valarray< typename _Dom::value_type >`
`&__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`

- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __shift_left, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __shift_left, typename _Dom1::value_type >::result_type > std::operator<< (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __shift_left, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< __shift_left, typename _Dom::value_type >::result_type > std::operator<< (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __shift_left, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __shift_left, typename _Dom::value_type >::result_type > std::operator<< (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __shift_left, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< __shift_left, typename _Dom::value_type >::result_type > std::operator<< (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __shift_left, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __shift_left, typename _Dom::value_type >::result_type > std::operator<< (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __less_equal, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __less_equal, typename _Dom1::value_type >::result_type > std::operator<= (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __less_equal, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun< __less_equal, typename _Dom::value_type >::result_type > std::operator<= (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __less_equal, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< __less_equal, typename _Dom::value_type >::result_type > std::operator<= (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __less_equal, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __less_equal, typename _Dom::value_type >::result_type > std::operator<= (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __less_equal, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __less_equal, typename _Dom::value_type >::result_type > std::operator<= (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< __equal_to, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun< __equal_to, typename _Dom::value_type >::result_type > std::operator== (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __equal_to, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __equal_to, typename _Dom::value_type >::result_type > std::operator== (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< __equal_to, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun< __equal_to, typename _Dom::value_type >::result_type > std::operator== (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`

- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __equal_to, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __equal_to, typename`
`_Dom1::value_type >::result_type > std::operator== (const _Expr< _Dom1, typename _Dom1::value_type >`
`&__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __equal_to, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun<`
`__equal_to, typename _Dom::value_type >::result_type > std::operator== (const _Expr< _Dom, typename`
`_Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __greater, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun<`
`__greater, typename _Dom::value_type >::result_type > std::operator> (const _Expr< _Dom, typename`
`_Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __greater, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun<`
`__greater, typename _Dom::value_type >::result_type > std::operator> (const _Expr< _Dom, typename`
`_Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __greater, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__greater, typename _Dom::value_type >::result_type > std::operator> (const valarray< typename`
`_Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __greater, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __greater, typename`
`_Dom1::value_type >::result_type > std::operator> (const _Expr< _Dom1, typename _Dom1::value_type >`
`&__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __greater, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__greater, typename _Dom::value_type >::result_type > std::operator> (const typename _Dom::value_type`
`&__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __greater_equal, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename`
`__fun< __greater_equal, typename _Dom::value_type >::result_type > std::operator>= (const _Expr< _Dom,`
`typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __greater_equal, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename`
`__fun< __greater_equal, typename _Dom::value_type >::result_type > std::operator>= (const typename`
`_Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __greater_equal, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename`
`__fun< __greater_equal, typename _Dom::value_type >::result_type > std::operator>= (const valarray< typename`
`_Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __greater_equal, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __greater_equal,`
`typename _Dom1::value_type >::result_type > std::operator>= (const _Expr< _Dom1, typename _Dom1::value`
`type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __greater_equal, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename`
`__fun< __greater_equal, typename _Dom::value_type >::result_type > std::operator>= (const _Expr< _Dom,`
`typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __shift_right, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __shift_right,`
`typename _Dom1::value_type >::result_type > std::operator>> (const _Expr< _Dom1, typename _Dom1::value`
`type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`

- `template<class _Dom >`
`_Expr< _BinClos< __shift_right, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun<`
`__shift_right, typename _Dom::value_type >::result_type > std::operator>> (const _Expr< _Dom, typename`
`_Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __shift_right, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__shift_right, typename _Dom::value_type >::result_type > std::operator>> (const typename _Dom::value_↵`
`type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __shift_right, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun<`
`__shift_right, typename _Dom::value_type >::result_type > std::operator>> (const _Expr< _Dom, typename`
`_Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __shift_right, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__shift_right, typename _Dom::value_type >::result_type > std::operator>> (const valarray< typename _↵`
`_Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< __bitwise_xor, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__bitwise_xor, typename _Dom::value_type >::result_type > std::operator^ (const valarray< typename _↵`
`_Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< __bitwise_xor, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun<`
`__bitwise_xor, typename _Dom::value_type >::result_type > std::operator^ (const _Expr< _Dom, typename`
`_Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __bitwise_xor, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __bitwise_xor, typename`
`_Dom1::value_type >::result_type > std::operator^ (const _Expr< _Dom1, typename _Dom1::value_type >`
`&__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __bitwise_xor, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun<`
`__bitwise_xor, typename _Dom::value_type >::result_type > std::operator^ (const _Expr< _Dom, typename`
`_Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __bitwise_xor, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__bitwise_xor, typename _Dom::value_type >::result_type > std::operator^ (const typename _Dom::value_↵`
`type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __bitwise_or, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __bitwise_or, typename`
`_Dom1::value_type >::result_type > std::operator| (const _Expr< _Dom1, typename _Dom1::value_type >`
`&__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< __bitwise_or, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun<`
`__bitwise_or, typename _Dom::value_type >::result_type > std::operator| (const _Expr< _Dom, typename ↵`
`_Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __bitwise_or, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun<`
`__bitwise_or, typename _Dom::value_type >::result_type > std::operator| (const _Expr< _Dom, typename ↵`
`_Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __bitwise_or, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__bitwise_or, typename _Dom::value_type >::result_type > std::operator| (const typename _Dom::value_type`
`&__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`

- `template<class _Dom >`
`_Expr< _BinClos< __bitwise_or, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__bitwise_or, typename _Dom::value_type >::result_type > std::operator| (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< __logical_or, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__logical_or, typename _Dom::value_type >::result_type > std::operator|| (const valarray< typename _Dom::value_type > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< __logical_or, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename __fun<`
`__logical_or, typename _Dom::value_type >::result_type > std::operator|| (const _Expr< _Dom, typename _Dom::value_type > &__v, const typename _Dom::value_type &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< __logical_or, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename __fun<`
`__logical_or, typename _Dom::value_type >::result_type > std::operator|| (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom >`
`_Expr< _BinClos< __logical_or, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename __fun<`
`__logical_or, typename _Dom::value_type >::result_type > std::operator|| (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__v)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< __logical_or, _Expr, _Expr, _Dom1, _Dom2 >, typename __fun< __logical_or, typename`
`_Dom1::value_type >::result_type > std::operator|| (const _Expr< _Dom1, typename _Dom1::value_type > &__v, const _Expr< _Dom2, typename _Dom2::value_type > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< _Pow, _ValArray, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_type`
`> std::pow (const valarray< typename _Dom::valarray > &__v, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _BinClos< _Pow, _Constant, _Expr, typename _Dom::value_type, _Dom >, typename _Dom::value_type`
`> std::pow (const typename _Dom::value_type &__t, const _Expr< _Dom, typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _BinClos< _Pow, _ValArray, _Constant, _Tp, _Tp >, _Tp > std::pow (const valarray< _Tp > &__v,`
`const _Tp &__t)`
- `template<class _Dom >`
`_Expr< _BinClos< _Pow, _Expr, _Constant, _Dom, typename _Dom::value_type >, typename _Dom::value_type`
`> std::pow (const _Expr< _Dom, typename _Dom::value_type > &__e, const typename _Dom::value_type &__t)`
- `template<typename _Tp >`
`_Expr< _BinClos< _Pow, _ValArray, _ValArray, _Tp, _Tp >, _Tp > std::pow (const valarray< _Tp > &__v,`
`const valarray< _Tp > &__w)`
- `template<class _Dom >`
`_Expr< _BinClos< _Pow, _Expr, _ValArray, _Dom, typename _Dom::value_type >, typename _Dom::value_type`
`> std::pow (const _Expr< _Dom, typename _Dom::value_type > &__e, const valarray< typename _Dom::value_type > &__v)`
- `template<class _Dom1, class _Dom2 >`
`_Expr< _BinClos< _Pow, _Expr, _Expr, _Dom1, _Dom2 >, typename _Dom1::value_type > std::pow (const`
`_Expr< _Dom1, typename _Dom1::value_type > &__e1, const _Expr< _Dom2, typename _Dom2::value_type > &__e2)`
- `template<typename _Tp >`
`_Expr< _BinClos< _Pow, _Constant, _ValArray, _Tp, _Tp >, _Tp > std::pow (const _Tp &__t, const valarray<`
`_Tp > &__v)`

- `template<class _Dom >`
`_Expr< _UnClos< _Sin, _Expr, _Dom >, typename _Dom::value_type > std::sin (const _Expr< _Dom, type-`
`name _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Sin, _ValArray, _Tp >, _Tp > std::sin (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< _Sinh, _Expr, _Dom >, typename _Dom::value_type > std::sinh (const _Expr< _Dom,`
`typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Sinh, _ValArray, _Tp >, _Tp > std::sinh (const valarray< _Tp > &__v)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Sqrt, _ValArray, _Tp >, _Tp > std::sqrt (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< _Sqrt, _Expr, _Dom >, typename _Dom::value_type > std::sqrt (const _Expr< _Dom, type-`
`name _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Tan, _ValArray, _Tp >, _Tp > std::tan (const valarray< _Tp > &__v)`
- `template<class _Dom >`
`_Expr< _UnClos< _Tan, _Expr, _Dom >, typename _Dom::value_type > std::tan (const _Expr< _Dom, type-`
`name _Dom::value_type > &__e)`
- `template<class _Dom >`
`_Expr< _UnClos< _Tanh, _Expr, _Dom >, typename _Dom::value_type > std::tanh (const _Expr< _Dom,`
`typename _Dom::value_type > &__e)`
- `template<typename _Tp >`
`_Expr< _UnClos< _Tanh, _ValArray, _Tp >, _Tp > std::tanh (const valarray< _Tp > &__v)`

6.666.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<valarray>`.

6.667 valarray_array.h File Reference

Namespaces

- [std](#)

Macros

- `#define _DEFINE_ARRAY_FUNCTION(_Op, _Name)`

Functions

- `template<typename _Tp >`
`void std::__valarray_copy (const _Tp *__restrict __a, size_t __n, _Tp *__restrict __b)`
- `template<typename _Tp >`
`void std::__valarray_copy (const _Tp *__restrict __a, size_t __n, size_t __s, _Tp *__restrict __b)`
- `template<typename _Tp >`
`void std::__valarray_copy (const _Tp *__restrict __a, _Tp *__restrict __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void std::__valarray_copy (const _Tp *__restrict __src, size_t __n, size_t __s1, _Tp *__restrict __dst, size_t __s2)`
- `template<typename _Tp >`
`void std::__valarray_copy (const _Tp *__restrict __a, const size_t *__restrict __i, _Tp *__restrict __b, size_t __n)`
- `template<typename _Tp >`
`void std::__valarray_copy (const _Tp *__restrict __a, size_t __n, _Tp *__restrict __b, const size_t *__restrict __i)`
- `template<typename _Tp >`
`void std::__valarray_copy (const _Tp *__restrict __src, size_t __n, const size_t *__restrict __i, _Tp *__restrict __dst, const size_t *__restrict __j)`
- `template<typename _Tp >`
`void std::__valarray_copy (_Array<_Tp> __a, size_t __n, _Array<_Tp> __b)`
- `template<typename _Tp >`
`void std::__valarray_copy (_Array<_Tp> __a, size_t __n, size_t __s, _Array<_Tp> __b)`
- `template<typename _Tp >`
`void std::__valarray_copy (_Array<_Tp> __a, _Array<_Tp> __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void std::__valarray_copy (_Array<_Tp> __a, size_t __n, size_t __s1, _Array<_Tp> __b, size_t __s2)`
- `template<typename _Tp >`
`void std::__valarray_copy (_Array<_Tp> __a, _Array<size_t> __i, _Array<_Tp> __b, size_t __n)`
- `template<typename _Tp >`
`void std::__valarray_copy (_Array<_Tp> __a, size_t __n, _Array<_Tp> __b, _Array<size_t> __i)`
- `template<typename _Tp >`
`void std::__valarray_copy (_Array<_Tp> __src, size_t __n, _Array<size_t> __i, _Array<_Tp> __dst, _Array<size_t> __j)`
- `template<typename _Tp >`
`void std::__valarray_copy_construct (const _Tp *__b, const _Tp *__e, _Tp *__restrict __o)`
- `template<typename _Tp >`
`void std::__valarray_copy_construct (const _Tp *__restrict __a, size_t __n, size_t __s, _Tp *__restrict __o)`
- `template<typename _Tp >`
`void std::__valarray_copy_construct (const _Tp *__restrict __a, const size_t *__restrict __i, _Tp *__restrict __o, size_t __n)`
- `template<typename _Tp >`
`void std::__valarray_copy_construct (_Array<_Tp> __a, _Array<size_t> __i, _Array<_Tp> __b, size_t __n)`
- `template<typename _Tp >`
`void std::__valarray_copy_construct (_Array<_Tp> __a, size_t __n, size_t __s, _Array<_Tp> __b)`
- `template<typename _Tp >`
`void std::__valarray_default_construct (_Tp *__b, _Tp *__e)`
- `template<typename _Tp >`
`void std::__valarray_destroy_elements (_Tp *__b, _Tp *__e)`

- `template<typename _Tp >`
`void std::__valarray_fill (_Tp *__restrict __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::__valarray_fill (_Tp *__restrict __a, size_t __n, size_t __s, const _Tp &__t)`
- `template<typename _Tp >`
`void std::__valarray_fill (_Tp *__restrict __a, const size_t *__restrict __i, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::__valarray_fill (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::__valarray_fill (_Array< _Tp > __a, size_t __n, size_t __s, const _Tp &__t)`
- `template<typename _Tp >`
`void std::__valarray_fill (_Array< _Tp > __a, _Array< size_t > __i, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::__valarray_fill_construct (_Tp *__b, _Tp *__e, const _Tp __t)`
- `void * std::__valarray_get_memory (size_t __n)`
- `template<typename _Tp >`
`_Tp *__restrict std::__valarray_get_storage (size_t __n)`
- `template<typename _Ta >`
`_Ta::value_type std::__valarray_max (const _Ta &__a)`
- `template<typename _Ta >`
`_Ta::value_type std::__valarray_min (const _Ta &__a)`
- `template<typename _Tp >`
`_Tp std::__valarray_product (const _Tp *__f, const _Tp *__l)`
- `void std::__valarray_release_memory (void *__p)`
- `template<typename _Tp >`
`_Tp std::__valarray_sum (const _Tp *__f, const _Tp *__l)`
- `template<typename _Tp, class _Dom >`
`void std::__Array_augmented_bitwise_and (_Array< _Tp > __a, _Array< size_t > __i, const Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::__Array_augmented_bitwise_and (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void std::__Array_augmented_bitwise_and (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp, class _Dom >`
`void std::__Array_augmented_bitwise_and (_Array< _Tp > __a, _Array< bool > __m, const Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::__Array_augmented_bitwise_and (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void std::__Array_augmented_bitwise_and (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::__Array_augmented_bitwise_and (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void std::__Array_augmented_bitwise_and (_Array< _Tp > __a, const Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::__Array_augmented_bitwise_and (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::__Array_augmented_bitwise_and (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`

- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__bitwise_and (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp >`
`&__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_and (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b,`
`size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__bitwise_or (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom,`
`_Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_or (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_or (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__bitwise_or (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t`
`__n)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_or (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_or (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__bitwise_or (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t`
`__n)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_or (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b,`
`size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__bitwise_or (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom,`
`_Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_or (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b,`
`size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_or (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array<`
`bool > __m)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_or (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array<`
`size_t > __i)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_xor (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_xor (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__bitwise_xor (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t`
`__n)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_xor (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::Array_augmented__bitwise_xor (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__bitwise_xor (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp >`
`&__e, size_t __n)`

- `template<typename _Tp >`
`void std::Array_augmented___bitwise_xor (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b,`
`size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented___bitwise_xor (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom,`
`_Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented___bitwise_xor (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom,`
`_Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented___bitwise_xor (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b,`
`size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented___bitwise_xor (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array<`
`bool > __m)`
- `template<typename _Tp >`
`void std::Array_augmented___bitwise_xor (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array<`
`size_t > __i)`
- `template<typename _Tp >`
`void std::Array_augmented___divides (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::Array_augmented___divides (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented___divides (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented___divides (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::Array_augmented___divides (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented___divides (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e,`
`size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented___divides (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b,`
`size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented___divides (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t`
`> __i)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented___divides (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp`
`> &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented___divides (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t`
`__n)`
- `template<typename _Tp >`
`void std::Array_augmented___divides (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool >`
`__m)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented___divides (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp`
`> &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented___minus (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp`
`> &__e, size_t __n)`

- `template<typename _Tp >`
`void std::Array_augmented__minus (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::Array_augmented__minus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::Array_augmented__minus (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__minus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__minus (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__minus (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__minus (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void std::Array_augmented__minus (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__minus (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__minus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp >`
`void std::Array_augmented__minus (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__modulus (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::Array_augmented__modulus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__modulus (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__modulus (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__modulus (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__modulus (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented__modulus (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__modulus (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented__modulus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`

- `template<typename _Tp, class _Dom >`
`void std::Array_augmented___modulus (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom,`
`_Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented___modulus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t >`
`__i > __i)`
- `template<typename _Tp >`
`void std::Array_augmented___modulus (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::Array_augmented___multiplies (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void std::Array_augmented___multiplies (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b,`
`size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented___multiplies (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array<`
`size_t > __i)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented___multiplies (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom,`
`_Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented___multiplies (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b,`
`size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented___multiplies (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool`
`> __m)`
- `template<typename _Tp >`
`void std::Array_augmented___multiplies (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented___multiplies (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented___multiplies (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom,`
`_Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented___multiplies (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e,`
`size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented___multiplies (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::Array_augmented___multiplies (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::Array_augmented___plus (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t`
`__n)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented___plus (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e,`
`size_t __n)`
- `template<typename _Tp >`
`void std::Array_augmented___plus (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::Array_augmented___plus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::Array_augmented___plus (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp, class _Dom >`
`void std::Array_augmented___plus (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`

- `template<typename _Tp >`
`void std::_Array_augmented__plus (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::_Array_augmented__plus (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::_Array_augmented__plus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp >`
`void std::_Array_augmented__plus (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp >`
`void std::_Array_augmented__plus (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp, class _Dom >`
`void std::_Array_augmented__plus (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::_Array_augmented__shift_left (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp, class _Dom >`
`void std::_Array_augmented__shift_left (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::_Array_augmented__shift_left (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::_Array_augmented__shift_left (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void std::_Array_augmented__shift_left (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::_Array_augmented__shift_left (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void std::_Array_augmented__shift_left (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp >`
`void std::_Array_augmented__shift_left (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp, class _Dom >`
`void std::_Array_augmented__shift_left (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::_Array_augmented__shift_left (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::_Array_augmented__shift_left (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< size_t > __i)`
- `template<typename _Tp, class _Dom >`
`void std::_Array_augmented__shift_left (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom, _Tp > &__e, size_t __n)`
- `template<typename _Tp >`
`void std::_Array_augmented__shift_right (_Array< _Tp > __a, _Array< _Tp > __b, size_t __n, size_t __s)`
- `template<typename _Tp >`
`void std::_Array_augmented__shift_right (_Array< _Tp > __a, size_t __n, size_t __s, _Array< _Tp > __b)`

- `template<typename _Tp >`
`void std::_Array_augmented__shift_right (_Array< _Tp > __a, _Array< size_t > __i, _Array< _Tp > __b,`
`size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::_Array_augmented__shift_right (_Array< _Tp > __a, _Array< bool > __m, const _Expr< _Dom,`
`_Tp > &__e, size_t __n)`
- `template<typename _Tp, class _Dom >`
`void std::_Array_augmented__shift_right (_Array< _Tp > __a, const _Expr< _Dom, _Tp > &__e, size_t`
`__n)`
- `template<typename _Tp >`
`void std::_Array_augmented__shift_right (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b,`
`size_t __n)`
- `template<typename _Tp >`
`void std::_Array_augmented__shift_right (_Array< _Tp > __a, size_t __n, const _Tp &__t)`
- `template<typename _Tp >`
`void std::_Array_augmented__shift_right (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array<`
`size_t > __i)`
- `template<typename _Tp, class _Dom >`
`void std::_Array_augmented__shift_right (_Array< _Tp > __a, size_t __s, const _Expr< _Dom, _Tp > &__e,`
`size_t __n)`
- `template<typename _Tp >`
`void std::_Array_augmented__shift_right (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b)`
- `template<typename _Tp >`
`void std::_Array_augmented__shift_right (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array<`
`bool > __m)`
- `template<typename _Tp, class _Dom >`
`void std::_Array_augmented__shift_right (_Array< _Tp > __a, _Array< size_t > __i, const _Expr< _Dom,`
`_Tp > &__e, size_t __n)`

6.667.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<valarray>`.

6.668 valarray_array.tcc File Reference

Namespaces

- [std](#)

Macros

- `#define _VALARRAY_ARRAY_TCC`

Functions

- `template<typename _Tp >`
`void std::__valarray_copy (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void std::__valarray_copy (_Array< _Tp > __a, size_t __n, _Array< _Tp > __b, _Array< bool > __m)`
- `template<typename _Tp >`
`void std::__valarray_copy (_Array< _Tp > __a, _Array< bool > __m, size_t __n, _Array< _Tp > __b, _Array< bool > __k)`
- `template<typename _Tp, class _Dom >`
`void std::__valarray_copy (const _Expr< _Dom, _Tp > &__e, size_t __n, _Array< _Tp > __a)`
- `template<typename _Tp, class _Dom >`
`void std::__valarray_copy (const _Expr< _Dom, _Tp > &__e, size_t __n, _Array< _Tp > __a, size_t __s)`
- `template<typename _Tp, class _Dom >`
`void std::__valarray_copy (const _Expr< _Dom, _Tp > &__e, size_t __n, _Array< _Tp > __a, _Array< size_t > __i)`
- `template<typename _Tp >`
`void std::__valarray_copy (_Array< _Tp > __e, _Array< size_t > __f, size_t __n, _Array< _Tp > __a, _Array< size_t > __i)`
- `template<typename _Tp, class _Dom >`
`void std::__valarray_copy (const _Expr< _Dom, _Tp > &__e, size_t __n, _Array< _Tp > __a, _Array< bool > __m)`
- `template<typename _Tp, class _Dom >`
`void std::__valarray_copy_construct (const _Expr< _Dom, _Tp > &__e, size_t __n, _Array< _Tp > __a)`
- `template<typename _Tp >`
`void std::__valarray_copy_construct (_Array< _Tp > __a, _Array< bool > __m, _Array< _Tp > __b, size_t __n)`
- `template<typename _Tp >`
`void std::__valarray_fill (_Array< _Tp > __a, size_t __n, _Array< bool > __m, const _Tp &__t)`

6.668.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<valarray>`.

6.669 valarray_before.h File Reference

Namespaces

- [std](#)

6.669.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<valarray>`.

6.670 vector File Reference

Macros

- `#define _GLIBCXX_VECTOR`

6.670.1 Detailed Description

This is a Standard C++ Library header.

6.671 vector File Reference

Classes

- class [__gnu_debug::__Safe_vector<_SafeSequence, _BaseSequence >](#)
- class [std::__debug::vector<_Tp, _Allocator >](#)
- struct [std::hash<__debug::vector<bool, _Alloc > >](#)

Namespaces

- [__gnu_debug](#)
- [std](#)
- [std::__debug](#)

Macros

- `#define _GLIBCXX_DEBUG_VECTOR`

Functions

- `template<typename _Tp, typename _Alloc >
bool std::__debug::operator!= (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool std::__debug::operator< (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool std::__debug::operator<= (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool std::__debug::operator== (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool std::__debug::operator> (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool std::__debug::operator>= (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
void std::__debug::swap (vector< _Tp, _Alloc > &__lhs, vector< _Tp, _Alloc > &__rhs) noexcept(/*conditional */)`

6.671.1 Detailed Description

This file is a GNU debug extension to the Standard C++ Library.

6.672 vector File Reference

Classes

- struct [std::hash< __profile::vector< bool, _Alloc > >](#)

Namespaces

- [std](#)
- [std::__profile](#)

Macros

- `#define _GLIBCXX_PROFILE_VECTOR`

Functions

- `template<typename _Tp, typename _Alloc >
bool std::__profile::operator!= (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool std::__profile::operator< (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool std::__profile::operator<= (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool std::__profile::operator== (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool std::__profile::operator> (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
bool std::__profile::operator>= (const vector< _Tp, _Alloc > &__lhs, const vector< _Tp, _Alloc > &__rhs)`
- `template<typename _Tp, typename _Alloc >
void std::__profile::swap (vector< _Tp, _Alloc > &__lhs, vector< _Tp, _Alloc > &__rhs) noexcept(*conditional
*)`

6.672.1 Detailed Description

This file is a GNU profile extension to the Standard C++ Library.

6.673 vector File Reference

Namespaces

- [std](#)

Macros

- `#define __cpp_lib_experimental_erase_if`
- `#define _GLIBCXX_EXPERIMENTAL_VECTOR`

Typedefs

- `template<typename _Tp >`
using **std::experimental::fundamentals_v2::pmr::vector** = `std::vector`<_Tp, polymorphic_allocator<_Tp >>

Functions

- `template<typename _Tp, typename _Alloc, typename _Up >`
void **std::experimental::fundamentals_v2::erase** (vector<_Tp, _Alloc > &__cont, const _Up &__value)
- `template<typename _Tp, typename _Alloc, typename _Predicate >`
void **std::experimental::fundamentals_v2::erase_if** (vector<_Tp, _Alloc > &__cont, _Predicate __pred)

6.673.1 Detailed Description

This is a TS C++ Library header.

6.674 vector.tcc File Reference

Namespaces

- `std`

Macros

- `#define _VECTOR_TCC`

6.674.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<vector>`.

6.675 vstring.h File Reference

Classes

- class `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base >`
- struct `std::hash<__gnu_cxx::__u16vstring >`
- struct `std::hash<__gnu_cxx::__u32vstring >`
- struct `std::hash<__gnu_cxx::__vstring >`
- struct `std::hash<__gnu_cxx::__wvstring >`

- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>
__versa_string< _CharT, _Traits, _Alloc, _Base > gnu_cxx::operator+ (__versa_string< _CharT, _Traits, ↵
_Alloc, _Base > &&__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>
__versa_string< _CharT, _Traits, _Alloc, _Base > gnu_cxx::operator+ (__versa_string< _CharT, _Traits, ↵
_Alloc, _Base > &&__lhs, _CharT __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>
bool gnu_cxx::operator< (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const __versa_↵
string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>
bool gnu_cxx::operator< (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const _CharT *↵
__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>
bool gnu_cxx::operator< (const _CharT *__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &↵
__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>
basic_ostream< _CharT, _Traits > & std::operator<< (basic_ostream< _CharT, _Traits > &__os, const __↵
gnu_cxx::__versa_string< _CharT, _Traits, _Alloc, _Base > &__str)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>
bool gnu_cxx::operator<= (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const __versa_↵
string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>
bool gnu_cxx::operator<= (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const _CharT ↵
*__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>
bool gnu_cxx::operator<= (const _CharT *__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > ↵
&__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>
bool gnu_cxx::operator== (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const __versa_↵
string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, template< typename, typename, typename > class _Base>
__enable_if< std::is_char< _CharT >::value, bool >::type gnu_cxx::operator== (const __versa_↵
string< _CharT, std::char_traits< _CharT >, std::allocator< _CharT >, _Base > &__lhs, const __versa_string< ↵
_CharT, std::char_traits< _CharT >, std::allocator< _CharT >, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>
bool gnu_cxx::operator== (const _CharT *__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &↵
__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>
bool gnu_cxx::operator== (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const _CharT *↵
__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>
bool gnu_cxx::operator> (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const __versa_↵
string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>
bool gnu_cxx::operator> (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const _CharT *↵
__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>
bool gnu_cxx::operator> (const _CharT *__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &↵
__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>
bool gnu_cxx::operator>= (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const __versa_↵
string< _CharT, _Traits, _Alloc, _Base > &__rhs)`

- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool __gnu_cxx::operator>= (const __versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, const _CharT`
`*__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`bool __gnu_cxx::operator>= (const _CharT *__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base >`
`&__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__is, __gnu_cxx::__↵`
`__versa_string< _CharT, _Traits, _Alloc, _Base > &__str)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`void __gnu_cxx::swap (__versa_string< _CharT, _Traits, _Alloc, _Base > &__lhs, __versa_string< _CharT, __↵`
`_Traits, _Alloc, _Base > &__rhs)`

6.675.1 Detailed Description

This file is a GNU extension to the Standard C++ Library.

6.676 vstring.tcc File Reference

Namespaces

- [__gnu_cxx](#)
- [std](#)

Macros

- `#define _VSTRING_TCC`

Functions

- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`basic_istream< _CharT, _Traits > & std::getline (basic_istream< _CharT, _Traits > &__is, __gnu_cxx::__↵`
`__string< _CharT, _Traits, _Alloc, _Base > &__str, _CharT __delim)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > __gnu_cxx::operator+ (const __versa_string< _CharT, _Traits,`
`_Alloc, _Base > &__lhs, const __versa_string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > __gnu_cxx::operator+ (const _CharT *__lhs, const __versa__↵`
`__string< _CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > __gnu_cxx::operator+ (_CharT __lhs, const __versa_string<`
`_CharT, _Traits, _Alloc, _Base > &__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > __gnu_cxx::operator+ (const __versa_string< _CharT, _Traits,`
`_Alloc, _Base > &__lhs, const _CharT *__rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`__versa_string< _CharT, _Traits, _Alloc, _Base > __gnu_cxx::operator+ (const __versa_string< _CharT, _Traits,`
`_Alloc, _Base > &__lhs, _CharT __rhs)`
- `template<typename _CharT, typename _Traits, typename _Alloc, template< typename, typename, typename > class _Base>`
`basic_istream< _CharT, _Traits > & std::operator>> (basic_istream< _CharT, _Traits > &__is, __gnu_cxx::__↵`
`__versa_string< _CharT, _Traits, _Alloc, _Base > &__str)`

6.676.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ext/vstring.h>`.

6.677 `vstring_fwd.h` File Reference

Classes

- class [__gnu_cxx::__rc_string_base<_CharT, _Traits, _Alloc>](#)
- class [__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _Base>](#)

Namespaces

- [__gnu_cxx](#)

Typedefs

- typedef `__versa_string< char, std::char_traits< char >, std::allocator< char >, __rc_string_base >` [__gnu_cxx::__rc_string](#)
- typedef `__vstring` [__gnu_cxx::__sso_string](#)
- typedef `__versa_string< char16_t, std::char_traits< char16_t >, std::allocator< char16_t >, __rc_string_base >` [__gnu_cxx::__u16rc_string](#)
- typedef `__u16vstring` [__gnu_cxx::__u16sso_string](#)
- typedef `__versa_string< char16_t >` [__gnu_cxx::__u16vstring](#)
- typedef `__versa_string< char32_t, std::char_traits< char32_t >, std::allocator< char32_t >, __rc_string_base >` [__gnu_cxx::__u32rc_string](#)
- typedef `__u32vstring` [__gnu_cxx::__u32sso_string](#)
- typedef `__versa_string< char32_t >` [__gnu_cxx::__u32vstring](#)
- typedef `__versa_string< char >` [__gnu_cxx::__vstring](#)
- typedef `__versa_string< wchar_t, std::char_traits< wchar_t >, std::allocator< wchar_t >, __rc_string_base >` [__gnu_cxx::__wrc_string](#)
- typedef `__wvstring` [__gnu_cxx::__wsso_string](#)
- typedef `__versa_string< wchar_t >` [__gnu_cxx::__wvstring](#)

6.677.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ext/vstring.h>`.

6.678 `vstring_util.h` File Reference

Namespaces

- [__gnu_cxx](#)

6.678.1 Detailed Description

This is an internal header file, included by other library headers. Do not attempt to use it directly. Instead, include `<ext/vstring.h>`.

6.679 workstealing.h File Reference

Classes

- struct [__gnu_parallel::__Job<_DifferenceTp>](#)

Namespaces

- [__gnu_parallel](#)

Macros

- `#define _GLIBCXX_JOB_VOLATILE`

Functions

- `template<typename _RAIter, typename _Op, typename _Fu, typename _Red, typename _Result>
_Op __gnu_parallel::__for_each_template_random_access_workstealing (_RAIter __begin, _RAIter __end, ↵
_Op __op, _Fu &__f, _Red __r, _Result __base, _Result &__output, typename std::iterator_traits<_RAIter>::↵
::difference_type __bound)`

6.679.1 Detailed Description

Parallelization of embarrassingly parallel execution by means of work-stealing.

Work stealing is described in

R. D. Blumofe and C. E. Leiserson. Scheduling multithreaded computations by work stealing. *Journal of the ACM*, 46(5):720–748, 1999.

This file is a GNU parallel extension to the Standard C++ Library.

Index

- `_AlgorithmStrategy`
 - `__gnu_parallel`, [409](#)
- `_BALLOC_ALIGN_BYTES`
 - `bitmap_allocator.h`, [3443](#)
- `_BinIndex`
 - `__gnu_parallel`, [409](#)
- `_Bit_scan_forward`
 - `__gnu_cxx`, [375](#)
- `_CASable`
 - `__gnu_parallel`, [409](#)
- `_CASable_bits`
 - `__gnu_parallel`, [450](#)
- `_CASable_mask`
 - `__gnu_parallel`, [450](#)
- `_Construct`
 - `std`, [570](#)
- `_DRandomShufflingGlobalData`
 - `__gnu_parallel::_DRandomShufflingGlobalData`, [1049](#)
- `_Destroy`
 - `std`, [570](#)
- `_Distance_precision`
 - `__gnu_debug`, [397](#)
- `_FindAlgorithm`
 - `__gnu_parallel`, [409](#)
- `_Find_first`
 - `SGI`, [303](#)
- `_Find_next`
 - `SGI`, [303](#)
- `_GLIBCXX_BAL_QUICKSORT`
 - `features.h`, [3529](#)
- `_GLIBCXX_CALL`
 - `compiletime_settings.h`, [3473](#)
- `_GLIBCXX_DEBUG_VERIFY_AT`
 - `macros.h`, [3607](#)
- `_GLIBCXX_DEQUE_BUF_SIZE`
 - `stl_deque.h`, [3769](#)
- `_GLIBCXX_FIND_CONSTANT_SIZE_BLOCKS`
 - `features.h`, [3529](#)
- `_GLIBCXX_FIND_EQUAL_SPLIT`
 - `features.h`, [3530](#)
- `_GLIBCXX_FIND_GROWING_BLOCKS`
 - `features.h`, [3530](#)
- `_GLIBCXX_MERGESORT`
 - `features.h`, [3530](#)
- `_GLIBCXX_PARALLEL_CONDITION`
 - `settings.h`, [3731](#)
- `_GLIBCXX_PARALLEL_LENGTH`
 - `multiway_merge.h`, [3626](#)
- `_GLIBCXX_PROFILE_DEFINE_UNINIT_DATA`
 - `__gnu_profile`, [457](#)
- `_GLIBCXX_QUICKSORT`
 - `features.h`, [3530](#)
- `_GLIBCXX_RANDOM_SHUFFLE_CONSIDER_L1`
 - `compiletime_settings.h`, [3474](#)
- `_GLIBCXX_RANDOM_SHUFFLE_CONSIDER_TLB`
 - `compiletime_settings.h`, [3474](#)
- `_GLIBCXX_SCALE_DOWN_FPU`
 - `compiletime_settings.h`, [3474](#)
- `_GLIBCXX_TREE_DYNAMIC_BALANCING`
 - `features.h`, [3530](#)
- `_GLIBCXX_TREE_FULL_COPY`
 - `features.h`, [3531](#)
- `_GLIBCXX_TREE_INITIAL_SPLITTING`
 - `features.h`, [3531](#)
- `_GLIBCXX_VERBOSE_LEVEL`
 - `compiletime_settings.h`, [3474](#)
- `_GLIBCXX_VOLATILE`
 - `partition.h`, [3653](#)
 - `queue.h`, [3678](#)
- `_GuardedIterator`
 - `__gnu_parallel::_GuardedIterator`, [1055](#)
- `_LoserTreeBase`
 - `__gnu_parallel::_LoserTreeBase`, [1073](#)
- `_M_allocate_and_copy`
 - `std::vector`, [3344](#)
- `_M_allocate_single_object`
 - `__gnu_cxx::bitmap_allocator`, [807](#)
- `_M_attach`
 - `__gnu_debug::Safe_forward_list`, [932](#)
 - `__gnu_debug::Safe_iterator`, [938](#)
 - `__gnu_debug::Safe_iterator_base`, [946](#)
 - `__gnu_debug::Safe_local_iterator`, [952](#)
 - `__gnu_debug::Safe_local_iterator_base`, [961](#)
 - `__gnu_debug::Safe_node_sequence`, [964](#)
 - `__gnu_debug::Safe_sequence`, [968](#)
 - `__gnu_debug::Safe_sequence_base`, [972](#)
 - `__gnu_debug::Safe_unordered_container`, [976](#)
 - `__gnu_debug::Safe_unordered_container_base`, [980](#)
 - `__gnu_debug::basic_string`, [989](#)
 - `std::__debug::deque`, [1453](#)
 - `std::__debug::forward_list`, [1457](#)
 - `std::__debug::list`, [1462](#)
 - `std::__debug::map`, [1467](#)
 - `std::__debug::multimap`, [1472](#)
 - `std::__debug::multiset`, [1477](#)
 - `std::__debug::set`, [1482](#)
 - `std::__debug::unordered_map`, [1487](#)
 - `std::__debug::unordered_multimap`, [1493](#)
 - `std::__debug::unordered_multiset`, [1498](#)
 - `std::__debug::unordered_set`, [1504](#)

- std::__debug::vector, 1510
- _M_attach_local
 - __gnu_debug::__Safe_unordered_container, 976
 - __gnu_debug::__Safe_unordered_container_base, 980
 - std::__debug::unordered_map, 1487
 - std::__debug::unordered_multimap, 1493
 - std::__debug::unordered_multiset, 1498
 - std::__debug::unordered_set, 1504
- _M_attach_local_single
 - __gnu_debug::__Safe_unordered_container, 976
 - __gnu_debug::__Safe_unordered_container_base, 980
 - std::__debug::unordered_map, 1487
 - std::__debug::unordered_multimap, 1493
 - std::__debug::unordered_multiset, 1498
 - std::__debug::unordered_set, 1504
- _M_attach_single
 - __gnu_debug::__Safe_forward_list, 932
 - __gnu_debug::__Safe_iterator, 938
 - __gnu_debug::__Safe_iterator_base, 946
 - __gnu_debug::__Safe_local_iterator, 952
 - __gnu_debug::__Safe_local_iterator_base, 961
 - __gnu_debug::__Safe_node_sequence, 964
 - __gnu_debug::__Safe_sequence, 968
 - __gnu_debug::__Safe_sequence_base, 972
 - __gnu_debug::__Safe_unordered_container, 976
 - __gnu_debug::__Safe_unordered_container_base, 980
 - __gnu_debug::basic_string, 989
 - std::__debug::deque, 1453
 - std::__debug::forward_list, 1457
 - std::__debug::list, 1462
 - std::__debug::map, 1467
 - std::__debug::multimap, 1472
 - std::__debug::multiset, 1477
 - std::__debug::set, 1482
 - std::__debug::unordered_map, 1487
 - std::__debug::unordered_multimap, 1493
 - std::__debug::unordered_multiset, 1499
 - std::__debug::unordered_set, 1504
 - std::__debug::vector, 1510
- _M_attached_to
 - __gnu_debug::__Safe_iterator, 938
 - __gnu_debug::__Safe_iterator_base, 946
 - __gnu_debug::__Safe_local_iterator, 952
 - __gnu_debug::__Safe_local_iterator_base, 961
- _M_before_dereferenceable
 - __gnu_debug::__Safe_iterator, 938
- _M_begin
 - __gnu_parallel::__Piece, 1090
- _M_bin_proc
 - __gnu_parallel::__DRandomShufflingGlobalData, 1049
- _M_bins_begin
 - __gnu_parallel::__DRSSorterPU, 1051
- _M_buf
 - __gnu_cxx::enc_filebuf, 832
 - __gnu_cxx::stdio_filebuf, 892
 - std::basic_filebuf, 1721
- _M_buf_locale
 - __gnu_cxx::enc_filebuf, 832
 - __gnu_cxx::stdio_filebuf, 892
 - __gnu_cxx::stdio_sync_filebuf, 913
 - std::basic_filebuf, 1721
 - std::basic_streambuf, 2164
 - std::basic_stringbuf, 2238
 - std::wbuffer_convert, 3377
- _M_buf_size
 - __gnu_cxx::enc_filebuf, 833
 - __gnu_cxx::stdio_filebuf, 892
 - std::basic_filebuf, 1721
- _M_can_compare
 - __gnu_debug::__Safe_iterator, 938
 - __gnu_debug::__Safe_iterator_base, 946
 - __gnu_debug::__Safe_local_iterator, 953
 - __gnu_debug::__Safe_local_iterator_base, 961
- _M_clear
 - __gnu_cxx::free_list, 839
- _M_comp
 - __gnu_parallel::__LoserTree, 1067
 - __gnu_parallel::__LoserTree< false, _Tp, _Compare >, 1071
 - __gnu_parallel::__LoserTreeBase, 1074
- _M_const_iterators
 - __gnu_debug::__Safe_forward_list, 934
 - __gnu_debug::__Safe_node_sequence, 966
 - __gnu_debug::__Safe_sequence, 970
 - __gnu_debug::__Safe_sequence_base, 974
 - __gnu_debug::__Safe_unordered_container, 978
 - __gnu_debug::__Safe_unordered_container_base, 982
 - __gnu_debug::basic_string, 1009
 - std::__debug::deque, 1454
 - std::__debug::forward_list, 1459
 - std::__debug::list, 1463
 - std::__debug::map, 1468
 - std::__debug::multimap, 1473
 - std::__debug::multiset, 1478
 - std::__debug::set, 1483
 - std::__debug::unordered_map, 1489
 - std::__debug::unordered_multimap, 1495
 - std::__debug::unordered_multiset, 1500
 - std::__debug::unordered_set, 1506
 - std::__debug::vector, 1512
- _M_const_local_iterators
 - __gnu_debug::__Safe_unordered_container, 978

- `__gnu_debug::Safe_unordered_container_base`, 982
 - `std::__debug::unordered_map`, 1489
 - `std::__debug::unordered_multimap`, 1495
 - `std::__debug::unordered_multiset`, 1500
 - `std::__debug::unordered_set`, 1506
- `_M_create_node`
 - `std::list`, 2738
- `_M_create_pback`
 - `__gnu_cxx::enc_filebuf`, 817
 - `__gnu_cxx::stdio_filebuf`, 874
 - `std::basic_filebuf`, 1703
- `_M_deallocate_single_object`
 - `__gnu_cxx::bitmap_allocator`, 807
- `_M_dereferenceable`
 - `__gnu_debug::Safe_iterator`, 939
 - `__gnu_debug::Safe_local_iterator`, 953
- `_M_destroy_pback`
 - `__gnu_cxx::enc_filebuf`, 817
 - `__gnu_cxx::stdio_filebuf`, 874
 - `std::basic_filebuf`, 1703
- `_M_detach`
 - `__gnu_debug::Safe_forward_list`, 932
 - `__gnu_debug::Safe_iterator`, 939
 - `__gnu_debug::Safe_iterator_base`, 946
 - `__gnu_debug::Safe_local_iterator`, 953
 - `__gnu_debug::Safe_local_iterator_base`, 961
 - `__gnu_debug::Safe_node_sequence`, 965
 - `__gnu_debug::Safe_sequence`, 968
 - `__gnu_debug::Safe_sequence_base`, 972
 - `__gnu_debug::Safe_unordered_container`, 976
 - `__gnu_debug::Safe_unordered_container_base`, 980
 - `__gnu_debug::basic_string`, 989
 - `std::__debug::deque`, 1453
 - `std::__debug::forward_list`, 1457
 - `std::__debug::list`, 1462
 - `std::__debug::map`, 1467
 - `std::__debug::multimap`, 1472
 - `std::__debug::multiset`, 1477
 - `std::__debug::set`, 1482
 - `std::__debug::unordered_map`, 1487
 - `std::__debug::unordered_multimap`, 1493
 - `std::__debug::unordered_multiset`, 1499
 - `std::__debug::unordered_set`, 1504
 - `std::__debug::vector`, 1510
- `_M_detach_all`
 - `__gnu_debug::Safe_forward_list`, 933
 - `__gnu_debug::Safe_node_sequence`, 965
 - `__gnu_debug::Safe_sequence`, 968
 - `__gnu_debug::Safe_sequence_base`, 973
 - `__gnu_debug::Safe_unordered_container`, 976
 - `__gnu_debug::Safe_unordered_container_base`, 981
- `__gnu_debug::basic_string`, 989
 - `std::__debug::deque`, 1453
 - `std::__debug::forward_list`, 1457
 - `std::__debug::list`, 1462
 - `std::__debug::map`, 1467
 - `std::__debug::multimap`, 1472
 - `std::__debug::multiset`, 1477
 - `std::__debug::set`, 1482
 - `std::__debug::unordered_map`, 1487
 - `std::__debug::unordered_multimap`, 1493
 - `std::__debug::unordered_multiset`, 1499
 - `std::__debug::unordered_set`, 1504
 - `std::__debug::vector`, 1510
- `_M_detach_local`
 - `__gnu_debug::Safe_unordered_container`, 976
 - `__gnu_debug::Safe_unordered_container_base`, 981
 - `std::__debug::unordered_map`, 1487
 - `std::__debug::unordered_multimap`, 1493
 - `std::__debug::unordered_multiset`, 1499
 - `std::__debug::unordered_set`, 1504
- `_M_detach_local_single`
 - `__gnu_debug::Safe_unordered_container`, 976
 - `__gnu_debug::Safe_unordered_container_base`, 981
 - `std::__debug::unordered_map`, 1487
 - `std::__debug::unordered_multimap`, 1493
 - `std::__debug::unordered_multiset`, 1499
 - `std::__debug::unordered_set`, 1504
- `_M_detach_single`
 - `__gnu_debug::Safe_forward_list`, 933
 - `__gnu_debug::Safe_iterator`, 939
 - `__gnu_debug::Safe_iterator_base`, 947
 - `__gnu_debug::Safe_local_iterator`, 953
 - `__gnu_debug::Safe_local_iterator_base`, 961
 - `__gnu_debug::Safe_node_sequence`, 965
 - `__gnu_debug::Safe_sequence`, 969
 - `__gnu_debug::Safe_sequence_base`, 973
 - `__gnu_debug::Safe_unordered_container`, 976
 - `__gnu_debug::Safe_unordered_container_base`, 981
 - `__gnu_debug::basic_string`, 989
 - `std::__debug::deque`, 1453
 - `std::__debug::forward_list`, 1457
 - `std::__debug::list`, 1462
 - `std::__debug::map`, 1467
 - `std::__debug::multimap`, 1472
 - `std::__debug::multiset`, 1477
 - `std::__debug::set`, 1482
 - `std::__debug::unordered_map`, 1488
 - `std::__debug::unordered_multimap`, 1493
 - `std::__debug::unordered_multiset`, 1499
 - `std::__debug::unordered_set`, 1505
 - `std::__debug::vector`, 1510

- `_M_detach_singular`
 - `__gnu_debug::__Safe_forward_list`, 933
 - `__gnu_debug::__Safe_node_sequence`, 965
 - `__gnu_debug::__Safe_sequence`, 969
 - `__gnu_debug::__Safe_sequence_base`, 973
 - `__gnu_debug::__Safe_unordered_container`, 976
 - `__gnu_debug::__Safe_unordered_container_base`, 981
 - `__gnu_debug::basic_string`, 989
 - `std::__debug::deque`, 1453
 - `std::__debug::forward_list`, 1458
 - `std::__debug::list`, 1462
 - `std::__debug::map`, 1467
 - `std::__debug::multimap`, 1472
 - `std::__debug::multiset`, 1477
 - `std::__debug::set`, 1482
 - `std::__debug::unordered_map`, 1488
 - `std::__debug::unordered_multimap`, 1493
 - `std::__debug::unordered_multiset`, 1499
 - `std::__debug::unordered_set`, 1505
 - `std::__debug::vector`, 1510
- `_M_dist`
 - `__gnu_parallel::__DRandomShufflingGlobalData`, 1049
- `_M_elements_leftover`
 - `__gnu_parallel::__QSBThreadLocal`, 1097
- `_M_end`
 - `__gnu_parallel::__Piece`, 1090
- `_M_ext_buf`
 - `__gnu_cxx::enc_filebuf`, 833
 - `__gnu_cxx::stdio_filebuf`, 892
 - `std::basic_filebuf`, 1721
- `_M_ext_buf_size`
 - `__gnu_cxx::enc_filebuf`, 833
 - `__gnu_cxx::stdio_filebuf`, 893
 - `std::basic_filebuf`, 1722
- `_M_ext_next`
 - `__gnu_cxx::enc_filebuf`, 833
 - `__gnu_cxx::stdio_filebuf`, 893
 - `std::basic_filebuf`, 1722
- `_M_fill_initialize`
 - `std::deque`, 2486
- `_M_finish_iterator`
 - `__gnu_parallel::__accumulate_selector`, 1011
 - `__gnu_parallel::__adjacent_difference_selector`, 1012
 - `__gnu_parallel::__count_if_selector`, 1018
 - `__gnu_parallel::__count_selector`, 1019
 - `__gnu_parallel::__fill_selector`, 1021
 - `__gnu_parallel::__for_each_selector`, 1025
 - `__gnu_parallel::__generate_selector`, 1027
 - `__gnu_parallel::__generic_for_each_selector`, 1030
 - `__gnu_parallel::__identity_selector`, 1031
 - `__gnu_parallel::__inner_product_selector`, 1033
 - `__gnu_parallel::__replace_if_selector`, 1041
 - `__gnu_parallel::__replace_selector`, 1043
 - `__gnu_parallel::__transform1_selector`, 1045
 - `__gnu_parallel::__transform2_selector`, 1046
- `_M_first`
 - `__gnu_parallel::__Job`, 1061
- `_M_first_insert`
 - `__gnu_parallel::__LoserTree`, 1068
 - `__gnu_parallel::__LoserTree< false, _Tp, _Compare >`, 1071
 - `__gnu_parallel::__LoserTreeBase`, 1074
- `_M_gcount`
 - `std::basic_fstream`, 1777
 - `std::basic_ifstream`, 1826
 - `std::basic_iostream`, 1913
 - `std::basic_istream`, 1960
 - `std::basic_istreamstringstream`, 2010
 - `std::basic_stringstream`, 2293
- `_M_get`
 - `__gnu_cxx::free_list`, 839
- `_M_get_mutex`
 - `__gnu_debug::__Safe_forward_list`, 933
 - `__gnu_debug::__Safe_iterator`, 939
 - `__gnu_debug::__Safe_iterator_base`, 947
 - `__gnu_debug::__Safe_local_iterator`, 953
 - `__gnu_debug::__Safe_local_iterator_base`, 961
 - `__gnu_debug::__Safe_node_sequence`, 965
 - `__gnu_debug::__Safe_sequence`, 969
 - `__gnu_debug::__Safe_sequence_base`, 973
 - `__gnu_debug::__Safe_unordered_container`, 977
 - `__gnu_debug::__Safe_unordered_container_base`, 981
 - `__gnu_debug::basic_string`, 989
 - `std::__debug::deque`, 1453
 - `std::__debug::forward_list`, 1458
 - `std::__debug::list`, 1462
 - `std::__debug::map`, 1467
 - `std::__debug::multimap`, 1472
 - `std::__debug::multiset`, 1477
 - `std::__debug::set`, 1482
 - `std::__debug::unordered_map`, 1488
 - `std::__debug::unordered_multimap`, 1494
 - `std::__debug::unordered_multiset`, 1499
 - `std::__debug::unordered_set`, 1505
 - `std::__debug::vector`, 1511
- `_M_get_result`
 - `std::__basic_future`, 1429
 - `std::future`, 2590
 - `std::future< _Res & >`, 2593
 - `std::future< void >`, 2595
 - `std::shared_future`, 3108
 - `std::shared_future< _Res & >`, 3111
 - `std::shared_future< void >`, 3113
- `_M_getloc`

- std::basic_fstream, 1735
- std::basic_ifstream, 1793
- std::basic_ios, 1840
- std::basic_iostream, 1870
- std::basic_istream, 1929
- std::basic_istreamstream, 1978
- std::basic_ofstream, 2025
- std::basic_ostream, 2065
- std::basic_ostreamstream, 2105
- std::basic_stringstream, 2251
- std::ios_base, 2659
- _M_global
 - __gnu_parallel::QSBThreadLocal, 1097
- _M_in_beg
 - __gnu_cxx::enc_filebuf, 833
 - __gnu_cxx::stdio_filebuf, 893
 - __gnu_cxx::stdio_sync_filebuf, 913
 - std::basic_filebuf, 1722
 - std::basic_streambuf, 2164
 - std::basic_stringbuf, 2238
 - std::wbuffer_convert, 3377
- _M_in_cur
 - __gnu_cxx::enc_filebuf, 833
 - __gnu_cxx::stdio_filebuf, 893
 - __gnu_cxx::stdio_sync_filebuf, 913
 - std::basic_filebuf, 1722
 - std::basic_streambuf, 2164
 - std::basic_stringbuf, 2239
 - std::wbuffer_convert, 3378
- _M_in_end
 - __gnu_cxx::enc_filebuf, 834
 - __gnu_cxx::stdio_filebuf, 893
 - __gnu_cxx::stdio_sync_filebuf, 913
 - std::basic_filebuf, 1722
 - std::basic_streambuf, 2164
 - std::basic_stringbuf, 2239
 - std::wbuffer_convert, 3378
- _M_in_same_bucket
 - __gnu_debug::Safe_local_iterator, 953
- _M_incrementable
 - __gnu_debug::Safe_iterator, 939
 - __gnu_debug::Safe_local_iterator, 954
- _M_initial
 - __gnu_parallel::QSBThreadLocal, 1097
- _M_initialize_map
 - std::Deque_base, 1602
 - std::deque, 2486
- _M_insert
 - __gnu_cxx::free_list, 840
- _M_invalidate
 - __gnu_debug::Safe_iterator, 939
 - __gnu_debug::Safe_iterator_base, 947
 - __gnu_debug::Safe_local_iterator, 954
 - __gnu_debug::Safe_local_iterator_base, 962
- _M_invalidate_all
 - __gnu_debug::Safe_forward_list, 933
 - __gnu_debug::Safe_node_sequence, 965
 - __gnu_debug::Safe_sequence, 969
 - __gnu_debug::Safe_sequence_base, 973
 - __gnu_debug::Safe_unordered_container, 977
 - __gnu_debug::Safe_unordered_container_base, 981
 - __gnu_debug::basic_string, 990
 - std::__debug::deque, 1453
 - std::__debug::forward_list, 1458
 - std::__debug::list, 1463
 - std::__debug::map, 1468
 - std::__debug::multimap, 1473
 - std::__debug::multiset, 1478
 - std::__debug::set, 1483
 - std::__debug::unordered_map, 1488
 - std::__debug::unordered_multimap, 1494
 - std::__debug::unordered_multiset, 1499
 - std::__debug::unordered_set, 1505
 - std::__debug::vector, 1511
- _M_invalidate_if
 - __gnu_debug::Safe_forward_list, 933
 - __gnu_debug::Safe_node_sequence, 965
 - __gnu_debug::Safe_sequence, 969
 - __gnu_debug::Safe_unordered_container, 977
 - __gnu_debug::basic_string, 990
 - std::__debug::deque, 1453
 - std::__debug::forward_list, 1458
 - std::__debug::list, 1463
 - std::__debug::map, 1468
 - std::__debug::multimap, 1473
 - std::__debug::multiset, 1478
 - std::__debug::set, 1483
 - std::__debug::unordered_map, 1488
 - std::__debug::unordered_multimap, 1494
 - std::__debug::unordered_multiset, 1500
 - std::__debug::unordered_set, 1505
 - std::__debug::vector, 1511
- _M_invalidate_local_if
 - __gnu_debug::Safe_unordered_container, 977
 - std::__debug::unordered_map, 1488
 - std::__debug::unordered_multimap, 1494
 - std::__debug::unordered_multiset, 1500
 - std::__debug::unordered_set, 1505
- _M_is_before_begin
 - __gnu_debug::Safe_iterator, 940
- _M_is_begin
 - __gnu_debug::Safe_iterator, 940
 - __gnu_debug::Safe_local_iterator, 954
- _M_is_beginnest
 - __gnu_debug::Safe_iterator, 940
- _M_is_end
 - __gnu_debug::Safe_iterator, 940

- __gnu_debug::Safe_local_iterator, 954
- _M_iterators
 - __gnu_debug::Safe_forward_list, 934
 - __gnu_debug::Safe_node_sequence, 966
 - __gnu_debug::Safe_sequence, 970
 - __gnu_debug::Safe_sequence_base, 974
 - __gnu_debug::Safe_unordered_container, 978
 - __gnu_debug::Safe_unordered_container_base, 982
 - __gnu_debug::basic_string, 1009
 - std::__debug::deque, 1454
 - std::__debug::forward_list, 1459
 - std::__debug::list, 1463
 - std::__debug::map, 1468
 - std::__debug::multimap, 1473
 - std::__debug::multiset, 1478
 - std::__debug::set, 1483
 - std::__debug::unordered_map, 1489
 - std::__debug::unordered_multimap, 1495
 - std::__debug::unordered_multiset, 1501
 - std::__debug::unordered_set, 1506
 - std::__debug::vector, 1512
- _M_key
 - __gnu_parallel::LoserTreeBase::_Loser, 1075
- _M_last
 - __gnu_parallel::Job, 1061
- _M_leftover_parts
 - __gnu_parallel::QSBThreadLocal, 1097
- _M_load
 - __gnu_parallel::Job, 1061
- _M_local_iterators
 - __gnu_debug::Safe_unordered_container, 978
 - __gnu_debug::Safe_unordered_container_base, 982
 - std::__debug::unordered_map, 1489
 - std::__debug::unordered_multimap, 1495
 - std::__debug::unordered_multiset, 1501
 - std::__debug::unordered_set, 1506
- _M_log_k
 - __gnu_parallel::LoserTree, 1068
 - __gnu_parallel::LoserTree< false, _Tp, _Compare >, 1071
 - __gnu_parallel::LoserTreeBase, 1074
- _M_losers
 - __gnu_parallel::LoserTree, 1068
 - __gnu_parallel::LoserTree< false, _Tp, _Compare >, 1071
 - __gnu_parallel::LoserTreeBase, 1074
- _M_mode
 - __gnu_cxx::enc_filebuf, 834
 - __gnu_cxx::stdio_filebuf, 893
 - std::basic_filebuf, 1722
 - std::basic_stringbuf, 2239
- _M_new_elements_at_back
 - std::deque, 2487
- _M_new_elements_at_front
 - std::deque, 2487
- _M_next
 - __gnu_debug::Safe_iterator, 943
 - __gnu_debug::Safe_iterator_base, 948
 - __gnu_debug::Safe_local_iterator, 958
 - __gnu_debug::Safe_local_iterator_base, 962
- _M_num_bins
 - __gnu_parallel::_DRandomShufflingGlobalData, 1049
- _M_num_bits
 - __gnu_parallel::_DRandomShufflingGlobalData, 1049
- _M_num_threads
 - __gnu_parallel::_DRSSorterPU, 1051
 - __gnu_parallel::_PMWMSortingData, 1092
 - __gnu_parallel::_QSBThreadLocal, 1097
- _M_offsets
 - __gnu_parallel::_PMWMSortingData, 1092
- _M_out_beg
 - __gnu_cxx::enc_filebuf, 834
 - __gnu_cxx::stdio_filebuf, 894
 - __gnu_cxx::stdio_sync_filebuf, 913
 - std::basic_filebuf, 1723
 - std::basic_streambuf, 2164
 - std::basic_stringbuf, 2239
 - std::wbuffer_convert, 3378
- _M_out_cur
 - __gnu_cxx::enc_filebuf, 834
 - __gnu_cxx::stdio_filebuf, 894
 - __gnu_cxx::stdio_sync_filebuf, 913
 - std::basic_filebuf, 1723
 - std::basic_streambuf, 2164
 - std::basic_stringbuf, 2239
 - std::wbuffer_convert, 3378
- _M_out_end
 - __gnu_cxx::enc_filebuf, 834
 - __gnu_cxx::stdio_filebuf, 894
 - __gnu_cxx::stdio_sync_filebuf, 914
 - std::basic_filebuf, 1723
 - std::basic_streambuf, 2165
 - std::basic_stringbuf, 2239
 - std::wbuffer_convert, 3378
- _M_pback
 - __gnu_cxx::enc_filebuf, 834
 - __gnu_cxx::stdio_filebuf, 894
 - std::basic_filebuf, 1723
- _M_pback_cur_save
 - __gnu_cxx::enc_filebuf, 834
 - __gnu_cxx::stdio_filebuf, 894
 - std::basic_filebuf, 1723
- _M_pback_end_save
 - __gnu_cxx::enc_filebuf, 835

- __gnu_cxx::stdio_filebuf, 895
 - std::basic_filebuf, 1724
- _M_pback_init
 - __gnu_cxx::enc_filebuf, 835
 - __gnu_cxx::stdio_filebuf, 895
 - std::basic_filebuf, 1724
- _M_pieces
 - __gnu_parallel::PMWMSSortingData, 1092
- _M_pop_back_aux
 - std::deque, 2487
- _M_pop_front_aux
 - std::deque, 2487
- _M_prior
 - __gnu_debug::Safe_iterator, 943
 - __gnu_debug::Safe_iterator_base, 948
 - __gnu_debug::Safe_local_iterator, 958
 - __gnu_debug::Safe_local_iterator_base, 962
- _M_push_back_aux
 - std::deque, 2488
- _M_push_front_aux
 - std::deque, 2488
- _M_range_check
 - std::deque, 2488
 - std::vector, 3344
- _M_range_initialize
 - std::deque, 2488, 2489
- _M_reading
 - __gnu_cxx::enc_filebuf, 835
 - __gnu_cxx::stdio_filebuf, 895
 - std::basic_filebuf, 1724
- _M_reallocate_map
 - std::deque, 2489
- _M_reserve_elements_at_back
 - std::deque, 2489
- _M_reserve_elements_at_front
 - std::deque, 2489
- _M_reserve_map_at_back
 - std::deque, 2490
- _M_reserve_map_at_front
 - std::deque, 2490
- _M_reset
 - __gnu_debug::Safe_iterator, 940
 - __gnu_debug::Safe_iterator_base, 947
 - __gnu_debug::Safe_local_iterator, 954
 - __gnu_debug::Safe_local_iterator_base, 962
- _M_revalidate_singular
 - __gnu_debug::Safe_forward_list, 933
 - __gnu_debug::Safe_node_sequence, 966
 - __gnu_debug::Safe_sequence, 969
 - __gnu_debug::Safe_sequence_base, 973
 - __gnu_debug::Safe_unordered_container, 977
 - __gnu_debug::Safe_unordered_container_base, 981
 - __gnu_debug::basic_string, 990
- std::__debug::deque, 1454
- std::__debug::forward_list, 1458
- std::__debug::list, 1463
- std::__debug::map, 1468
- std::__debug::multimap, 1473
- std::__debug::multiset, 1478
- std::__debug::set, 1483
- std::__debug::unordered_map, 1488
- std::__debug::unordered_multimap, 1494
- std::__debug::unordered_multiset, 1500
- std::__debug::unordered_set, 1505
- std::__debug::vector, 1511
- _M_samples
 - __gnu_parallel::PMWMSSortingData, 1093
- _M_sd
 - __gnu_parallel::DRSSorterPU, 1051
- _M_seed
 - __gnu_parallel::DRSSorterPU, 1051
- _M_sequence
 - __gnu_debug::Safe_iterator, 944
 - __gnu_debug::Safe_iterator_base, 948
 - __gnu_debug::Safe_local_iterator, 958
 - __gnu_debug::Safe_local_iterator_base, 963
- _M_sequential_algorithm
 - __gnu_parallel::__adjacent_find_selector, 1013
 - __gnu_parallel::__find_first_of_selector, 1022
 - __gnu_parallel::__find_if_selector, 1023
 - __gnu_parallel::__mismatch_selector, 1036
- _M_set_buffer
 - __gnu_cxx::enc_filebuf, 817
 - __gnu_cxx::stdio_filebuf, 874
 - std::basic_filebuf, 1703
- _M_set_node
 - std::Deque_iterator, 1604
- _M_singular
 - __gnu_debug::Safe_iterator, 940
 - __gnu_debug::Safe_iterator_base, 947
 - __gnu_debug::Safe_local_iterator, 955
 - __gnu_debug::Safe_local_iterator_base, 962
- _M_source
 - __gnu_parallel::DRandomShufflingGlobalData, 1049
 - __gnu_parallel::LoserTreeBase::Loser, 1075
 - __gnu_parallel::PMWMSSortingData, 1093
- _M_starts
 - __gnu_parallel::DRandomShufflingGlobalData, 1050
 - __gnu_parallel::PMWMSSortingData, 1093
- _M_sup
 - __gnu_parallel::LoserTreeBase::Loser, 1076
- _M_swap
 - __gnu_debug::Safe_node_sequence, 966
 - __gnu_debug::Safe_sequence, 969
 - __gnu_debug::Safe_sequence_base, 973

- `__gnu_debug::Safe_unordered_container`, 977, 978
 - `__gnu_debug::Safe_unordered_container_base`, 982
 - `__gnu_debug::basic_string`, 990
 - `std::__debug::deque`, 1454
 - `std::__debug::list`, 1463
 - `std::__debug::map`, 1468
 - `std::__debug::multimap`, 1473
 - `std::__debug::multiset`, 1478
 - `std::__debug::set`, 1483
 - `std::__debug::unordered_map`, 1489
 - `std::__debug::unordered_multimap`, 1494
 - `std::__debug::unordered_multiset`, 1500
 - `std::__debug::unordered_set`, 1506
 - `std::__debug::vector`, 1511
- `_M_temporaries`
 - `__gnu_parallel::DRandomShufflingGlobalData`, 1050
- `_M_temporary`
 - `__gnu_parallel::PMWMSSortingData`, 1093
- `_M_transfer_from_if`
 - `__gnu_debug::Safe_forward_list`, 933
 - `__gnu_debug::Safe_node_sequence`, 966
 - `__gnu_debug::Safe_sequence`, 970
 - `__gnu_debug::basic_string`, 990
 - `std::__debug::deque`, 1454
 - `std::__debug::forward_list`, 1458
 - `std::__debug::list`, 1463
 - `std::__debug::map`, 1468
 - `std::__debug::multimap`, 1473
 - `std::__debug::multiset`, 1478
 - `std::__debug::set`, 1483
 - `std::__debug::vector`, 1511
- `_M_unlink`
 - `__gnu_debug::Safe_iterator`, 941
 - `__gnu_debug::Safe_iterator_base`, 947
 - `__gnu_debug::Safe_local_iterator`, 955
 - `__gnu_debug::Safe_local_iterator_base`, 962
- `_M_use_pointer`
 - `__gnu_parallel::LoserTreeTraits`, 1084
- `_M_version`
 - `__gnu_debug::Safe_forward_list`, 934
 - `__gnu_debug::Safe_iterator`, 944
 - `__gnu_debug::Safe_iterator_base`, 948
 - `__gnu_debug::Safe_local_iterator`, 958
 - `__gnu_debug::Safe_local_iterator_base`, 963
 - `__gnu_debug::Safe_node_sequence`, 967
 - `__gnu_debug::Safe_sequence`, 970
 - `__gnu_debug::Safe_sequence_base`, 974
 - `__gnu_debug::Safe_unordered_container`, 978
 - `__gnu_debug::Safe_unordered_container_base`, 982
 - `__gnu_debug::basic_string`, 1009
 - `std::__debug::deque`, 1454
 - `std::__debug::forward_list`, 1459
 - `std::__debug::list`, 1464
 - `std::__debug::map`, 1469
 - `std::__debug::multimap`, 1474
 - `std::__debug::multiset`, 1479
 - `std::__debug::set`, 1484
 - `std::__debug::unordered_map`, 1489
 - `std::__debug::unordered_multimap`, 1495
 - `std::__debug::unordered_multiset`, 1501
 - `std::__debug::unordered_set`, 1506
 - `std::__debug::vector`, 1512
- `_M_w`
 - `std::Base_bitset`, 1598
 - `std::tr2::dynamic_bitset_base`, 3184
- `_M_write`
 - `std::basic_fstream`, 1735
 - `std::basic_iostream`, 1870
 - `std::basic_ofstream`, 2025
 - `std::basic_ostream`, 2065
 - `std::basic_ostringstream`, 2105
 - `std::basic_stringstream`, 2251
- `_MultiwayMergeAlgorithm`
 - `__gnu_parallel`, 410
- `_Opcode`
 - Base and Implementation Classes, 25
- `_PCCP`
 - `__gnu_parallel::IteratorPair`, 1058
 - `std::pair`, 3013
 - `std::sub_match`, 3140
- `_PCCFP`
 - `__gnu_parallel::IteratorPair`, 1058
 - `std::pair`, 3013
 - `std::sub_match`, 3140
- `_Parallelism`
 - `__gnu_parallel`, 410
- `_PartialSumAlgorithm`
 - `__gnu_parallel`, 410
- `_Piece`
 - `__gnu_parallel::QSBThreadLocal`, 1096
- `_PseudoSequence`
 - `__gnu_parallel::PseudoSequence`, 1094
- `_Ptr`
 - `std::__basic_future`, 1429
 - `std::__future_base`, 1559
 - `std::future`, 2590
 - `std::future< _Res & >`, 2592
 - `std::future< void >`, 2595
 - `std::shared_future`, 3108
 - `std::shared_future< _Res & >`, 3110
 - `std::shared_future< void >`, 3113
- `_QSBThreadLocal`
 - `__gnu_parallel::QSBThreadLocal`, 1096
- `_RandomNumber`
 - `__gnu_parallel::RandomNumber`, 1098

- `_RestrictedBoundedConcurrentQueue`
 - `__gnu_parallel::_RestrictedBoundedConcurrentQueue`, 1100
- `_Safe_iterator`
 - `__gnu_debug::_Safe_iterator`, 936, 937
- `_Safe_iterator_base`
 - `__gnu_debug::_Safe_iterator_base`, 945, 946
- `_Safe_local_iterator`
 - `__gnu_debug::_Safe_local_iterator`, 951
- `_Safe_local_iterator_base`
 - `__gnu_debug::_Safe_local_iterator_base`, 960
- `_SequenceIndex`
 - `__gnu_parallel`, 409
- `_SortAlgorithm`
 - `__gnu_parallel`, 410
- `_SplittingAlgorithm`
 - `__gnu_parallel`, 410
- `_Temporary_buffer`
 - `std::Temporary_buffer`, 1634
- `_ThreadIndex`
 - `__gnu_parallel`, 409
- `_TokenT`
 - `std::detail::Scanner`, 1556
- `_Unchecked_flip`
 - SGI, 304
- `_Unchecked_reset`
 - SGI, 304
- `_Unchecked_set`
 - SGI, 304
- `_Unchecked_test`
 - SGI, 304
- `__addressof`
 - Utilities, 353
- `__allocate_guarded`
 - std, 562
- `__allocated_ptr`
 - `std::allocated_ptr`, 1424
- `__allocator_base`
 - Allocators, 10
- `__begin1_iterator`
 - `__gnu_parallel::inner_product_selector`, 1033
- `__begin2_iterator`
 - `__gnu_parallel::inner_product_selector`, 1033
- `__bins_end`
 - `__gnu_parallel::DRSSorterPU`, 1051
- `__bit_allocate`
 - `__gnu_cxx::detail`, 389
- `__bit_free`
 - `__gnu_cxx::detail`, 389
- `__calc_borders`
 - `__gnu_parallel`, 410
- `__check_dereferenceable`
 - `__gnu_debug`, 397, 398
- `__check_singular`
 - `__gnu_debug`, 398
- `__check_singular_aux`
 - `__gnu_debug`, 398
- `__check_string`
 - `__gnu_debug`, 398
- `__compare_and_swap`
 - `__gnu_parallel`, 411
- `__constexpr_addressof`
 - optional, 3646
 - Optional values, 230
- `__cpp_lib_shared_timed_mutex`
 - Mutexes, 169
- `__ctype_type`
 - `std::basic_ios`, 1836
- `__cxa_demangle`
 - `cxxabi.h`, 3498
- `__cxxabiv1::__forced_unwind`, 712
- `__decode2`
 - `__gnu_parallel`, 411
- `__delete_min_insert`
 - `__gnu_parallel::LoserTree`, 1067
 - `__gnu_parallel::LoserTree< false, _Tp, _Compare >`, 1069
- `__determine_samples`
 - `__gnu_parallel`, 412
- `__encode2`
 - `__gnu_parallel`, 412
- `__env_t`
 - `__gnu_profile`, 457
- `__equally_split`
 - `__gnu_parallel`, 412
- `__equally_split_point`
 - `__gnu_parallel`, 413
- `__fetch_and_add`
 - `__gnu_parallel`, 413
- `__final_insertion_sort`
 - std, 562
- `__find_if`
 - std, 562
- `__find_if_not`
 - std, 563
- `__find_if_not_n`
 - std, 563
- `__find_template`
 - `__gnu_parallel`, 414, 415
- `__for_each_template_random_access`
 - `__gnu_parallel`, 416
- `__for_each_template_random_access_ed`
 - `__gnu_parallel`, 416
- `__for_each_template_random_access_omp_loop`
 - `__gnu_parallel`, 417
- `__for_each_template_random_access_omp_loop_static`
 - `__gnu_parallel`, 417
- `__for_each_template_random_access_workstealing`

- __gnu_parallel, 418
- __foreign_iterator_aux2
- __gnu_debug, 398, 399
- __gcd
 - std, 563
- __genrand_bits
 - __gnu_parallel::RandomNumber, 1098
- __get_distance
 - __gnu_debug, 399
- __get_min_source
 - __gnu_parallel::LoserTree, 1067
 - __gnu_parallel::LoserTree< false, _Tp, _Compare >, 1070
 - __gnu_parallel::LoserTreeBase, 1073
- __get_num_threads
 - __gnu_parallel::balanced_quicksort_tag, 1111
 - __gnu_parallel::balanced_tag, 1112
 - __gnu_parallel::default_parallel_tag, 1114
 - __gnu_parallel::exact_tag, 1116
 - __gnu_parallel::multiway_mergesort_exact_tag, 1119
 - __gnu_parallel::multiway_mergesort_sampling_tag, 1120
 - __gnu_parallel::multiway_mergesort_tag, 1122
 - __gnu_parallel::omp_loop_static_tag, 1123
 - __gnu_parallel::omp_loop_tag, 1124
 - __gnu_parallel::parallel_tag, 1127
 - __gnu_parallel::quicksort_tag, 1129
 - __gnu_parallel::sampling_tag, 1130
 - __gnu_parallel::unbalanced_tag, 1131
- __glibcxx_check_erase
 - macros.h, 3605
- __glibcxx_check_erase_after
 - macros.h, 3605
- __glibcxx_check_erase_range
 - macros.h, 3605
- __glibcxx_check_erase_range_after
 - macros.h, 3605
- __glibcxx_check_heap_pred
 - macros.h, 3605
- __glibcxx_check_insert
 - macros.h, 3605
- __glibcxx_check_insert_after
 - macros.h, 3605
- __glibcxx_check_insert_range
 - macros.h, 3606
- __glibcxx_check_insert_range_after
 - macros.h, 3606
- __glibcxx_check_partitioned_lower
 - macros.h, 3606
- __glibcxx_check_partitioned_lower_pred
 - macros.h, 3606
- __glibcxx_check_partitioned_upper_pred
 - macros.h, 3606
- __glibcxx_check_sorted_pred
 - macros.h, 3606
- __gnu_cxx, 360
 - __Bit_scan_forward, 375
 - __static_pointer_cast, 375
 - conf_hyperg, 375
 - conf_hypergf, 376
 - conf_hypergl, 376
 - hyperg, 376
 - hypergf, 378
 - hypergl, 378
 - operator!=, 378, 379
 - operator<, 381, 382
 - operator<=, 383
 - operator>, 385, 386
 - operator>=, 386, 387
 - operator+, 379–381
 - operator==, 384, 385
 - swap, 388
- __gnu_cxx::Caster< _ToType >, 794
- __gnu_cxx::Char_types< _CharT >, 794
- __gnu_cxx::ExtPtr_allocator< _Tp >, 795
- __gnu_cxx::Invalid_type, 796
- __gnu_cxx::Pointer_adapter< _Storage_policy >, 796
- __gnu_cxx::Relative_pointer_impl< _Tp >, 798
- __gnu_cxx::Relative_pointer_impl< const _Tp >, 799
- __gnu_cxx::Std_pointer_impl< _Tp >, 800
- __gnu_cxx::Unqualified_type< _Tp >, 800
- __gnu_cxx::__alloc_traits
 - allocate, 716
 - const_void_pointer, 715
 - construct, 717
 - deallocate, 717
 - destroy, 717
 - is_always_equal, 715
 - max_size, 718
 - propagate_on_container_copy_assignment, 715
 - propagate_on_container_move_assignment, 715
 - propagate_on_container_swap, 715
 - select_on_container_copy_construction, 718
 - void_pointer, 716
- __gnu_cxx::__alloc_traits< _Alloc >, 713
- __gnu_cxx::__common_pool_policy< _PoolTp, _Thread >, 719
- __gnu_cxx::__detail, 388
 - __bit_allocate, 389
 - __bit_free, 389
 - __num_bitmaps, 389
 - __num_blocks, 389
- __gnu_cxx::__detail::Bitmap_counter< _Tp >, 720
- __gnu_cxx::__detail::Ffit_finder
 - argument_type, 721
 - result_type, 721
- __gnu_cxx::__detail::Ffit_finder< _Tp >, 721

- `__gnu_cxx::__detail::__mini_vector<_Tp>`, 719
- `__gnu_cxx::__mt_alloc<_Tp, _Poolp>`, 722
- `__gnu_cxx::__mt_alloc_base<_Tp>`, 723
- `__gnu_cxx::__per_type_pool_policy<_Tp, _PoolTp, _←
Thread>`, 724
- `__gnu_cxx::__pool<_Thread>`, 725
- `__gnu_cxx::__pool<false>`, 725
- `__gnu_cxx::__pool<true>`, 726
- `__gnu_cxx::__pool_alloc<_Tp>`, 727
- `__gnu_cxx::__pool_alloc_base`, 729
- `__gnu_cxx::__pool_base`, 730
- `__gnu_cxx::__rc_string_base<_CharT, _Traits, _Alloc>`,
731
- `__gnu_cxx::__scoped_lock`, 734
- `__gnu_cxx::__versa_string`
 - `__versa_string`, 738–741
 - `~__versa_string`, 741
 - `append`, 741, 742, 744, 745
 - `assign`, 745–748
 - `at`, 749
 - `back`, 750
 - `begin`, 750
 - `c_str`, 751
 - `capacity`, 751
 - `cbegin`, 751
 - `cend`, 751
 - `clear`, 751
 - `compare`, 751–754
 - `copy`, 755
 - `crbegin`, 755
 - `crend`, 755
 - `data`, 756
 - `empty`, 756
 - `end`, 756
 - `erase`, 757, 758
 - `find`, 758, 759
 - `find_first_not_of`, 760, 761
 - `find_first_of`, 762, 763
 - `find_last_not_of`, 764, 765
 - `find_last_of`, 766, 767
 - `front`, 768
 - `get_allocator`, 768
 - `insert`, 768–772, 774
 - `length`, 774
 - `max_size`, 775
 - `npos`, 793
 - `operator+=`, 775, 776
 - `operator=`, 776–778
 - `operator[]`, 778, 779
 - `pop_back`, 779
 - `push_back`, 779
 - `rbegin`, 780
 - `rend`, 780
 - `replace`, 780–785, 787, 788
 - `reserve`, 788
 - `resize`, 789
 - `rfind`, 790, 791
 - `shrink_to_fit`, 792
 - `size`, 792
 - `substr`, 792
 - `swap`, 793
- `__gnu_cxx::__versa_string<_CharT, _Traits, _Alloc, _←
Base>`, 734
- `__gnu_cxx::annotate_base`, 801
- `__gnu_cxx::array_allocator<_Tp, _Array>`, 802
- `__gnu_cxx::array_allocator_base<_Tp>`, 803
- `__gnu_cxx::binary_compose`
 - `argument_type`, 805
 - `result_type`, 805
- `__gnu_cxx::binary_compose<_Operation1, _Operation2,
_Operation3>`, 804
- `__gnu_cxx::bitmap_allocator`
 - `_M_allocate_single_object`, 807
 - `_M_deallocate_single_object`, 807
- `__gnu_cxx::bitmap_allocator<_Tp>`, 806
- `__gnu_cxx::char_traits<_CharT>`, 808
- `__gnu_cxx::character<_Value, _Int, _St>`, 809
- `__gnu_cxx::condition_base`, 810
- `__gnu_cxx::constant_binary_fun<_Result, _Arg1, _Arg2
>`, 811
- `__gnu_cxx::constant_unary_fun<_Result, _Argument>`,
811
- `__gnu_cxx::constant_void_fun<_Result>`, 812
- `__gnu_cxx::debug_allocator<_Alloc>`, 813
- `__gnu_cxx::enc_filebuf`
 - `_M_buf`, 832
 - `_M_buf_locale`, 832
 - `_M_buf_size`, 833
 - `_M_create_pback`, 817
 - `_M_destroy_pback`, 817
 - `_M_ext_buf`, 833
 - `_M_ext_buf_size`, 833
 - `_M_ext_next`, 833
 - `_M_in_beg`, 833
 - `_M_in_cur`, 833
 - `_M_in_end`, 834
 - `_M_mode`, 834
 - `_M_out_beg`, 834
 - `_M_out_cur`, 834
 - `_M_out_end`, 834
 - `_M_pback`, 834
 - `_M_pback_cur_save`, 834
 - `_M_pback_end_save`, 835
 - `_M_pback_init`, 835
 - `_M_reading`, 835
 - `_M_set_buffer`, 817
- `close`, 817
- `eback`, 817

- egptr, [818](#)
- epptr, [818](#)
- gbump, [818](#)
- getloc, [819](#)
- gptr, [819](#)
- imbue, [819](#)
- in_avail, [820](#)
- is_open, [820](#)
- open, [820](#), [821](#)
- overflow, [821](#)
- pbackfail, [822](#)
- pbase, [822](#)
- pbump, [823](#)
- pptr, [823](#)
- pubimbue, [823](#)
- pubseekoff, [824](#)
- pubseekpos, [824](#)
- pubsetbuf, [824](#)
- pubsync, [825](#)
- sbumpc, [825](#)
- seekoff, [825](#)
- seekpos, [825](#)
- setbuf, [826](#)
- setg, [826](#)
- setp, [827](#)
- sgetc, [827](#)
- sgetn, [827](#)
- showmanyc, [828](#)
- snextc, [828](#)
- sputbackc, [828](#)
- sputc, [829](#)
- sputn, [829](#)
- sungetc, [830](#)
- sync, [830](#)
- uflow, [830](#)
- underflow, [831](#)
- xsggetn, [831](#)
- xsgputn, [832](#)
- __gnu_cxx::enc_filebuf<_CharT>, [814](#)
- __gnu_cxx::encoding_char_traits<_CharT>, [836](#)
- __gnu_cxx::encoding_state, [837](#)
- __gnu_cxx::forced_error, [838](#)
 - what, [838](#)
- __gnu_cxx::free_list, [839](#)
 - _M_clear, [839](#)
 - _M_get, [839](#)
 - _M_insert, [840](#)
- __gnu_cxx::hash_map<_Key, _Tp, _HashFn, _EqualKey, _Alloc>, [840](#)
- __gnu_cxx::hash_multimap<_Key, _Tp, _HashFn, _EqualKey, _Alloc>, [842](#)
- __gnu_cxx::hash_multiset<_Value, _HashFn, _EqualKey, _Alloc>, [844](#)
- __gnu_cxx::hash_set<_Value, _HashFn, _EqualKey, _Alloc>, [846](#)
- __gnu_cxx::limit_condition, [847](#)
- __gnu_cxx::limit_condition::always_adjustor, [848](#)
- __gnu_cxx::limit_condition::limit_adjustor, [848](#)
- __gnu_cxx::limit_condition::never_adjustor, [849](#)
- __gnu_cxx::malloc_allocator<_Tp>, [849](#)
- __gnu_cxx::new_allocator<_Tp>, [850](#)
- __gnu_cxx::project1st
 - first_argument_type, [852](#)
 - result_type, [852](#)
 - second_argument_type, [852](#)
- __gnu_cxx::project1st<_Arg1, _Arg2>, [852](#)
- __gnu_cxx::project2nd
 - first_argument_type, [853](#)
 - result_type, [853](#)
 - second_argument_type, [853](#)
- __gnu_cxx::project2nd<_Arg1, _Arg2>, [853](#)
- __gnu_cxx::random_condition, [854](#)
- __gnu_cxx::random_condition::always_adjustor, [854](#)
- __gnu_cxx::random_condition::group_adjustor, [855](#)
- __gnu_cxx::random_condition::never_adjustor, [855](#)
- __gnu_cxx::rb_tree<_Key, _Value, _KeyOfValue, _Compare, _Alloc>, [856](#)
- __gnu_cxx::recursive_init_error, [859](#)
 - what, [860](#)
- __gnu_cxx::rope<_CharT, _Alloc>, [860](#)
- __gnu_cxx::select1st
 - argument_type, [866](#)
 - result_type, [866](#)
- __gnu_cxx::select1st<_Pair>, [865](#)
- __gnu_cxx::select2nd
 - argument_type, [867](#)
 - result_type, [867](#)
- __gnu_cxx::select2nd<_Pair>, [866](#)
- __gnu_cxx::slist<_Tp, _Alloc>, [867](#)
- __gnu_cxx::stdio_filebuf
 - _M_buf, [892](#)
 - _M_buf_locale, [892](#)
 - _M_buf_size, [892](#)
 - _M_create_pback, [874](#)
 - _M_destroy_pback, [874](#)
 - _M_ext_buf, [892](#)
 - _M_ext_buf_size, [893](#)
 - _M_ext_next, [893](#)
 - _M_in_beg, [893](#)
 - _M_in_cur, [893](#)
 - _M_in_end, [893](#)
 - _M_mode, [893](#)
 - _M_out_beg, [894](#)
 - _M_out_cur, [894](#)
 - _M_out_end, [894](#)
 - _M_pback, [894](#)
 - _M_pback_cur_save, [894](#)

- [_M_pback_end_save, 895](#)
- [_M_pback_init, 895](#)
- [_M_reading, 895](#)
- [_M_set_buffer, 874](#)
- [~stdio_filebuf, 874](#)
- [close, 875](#)
- [eback, 875](#)
- [egptr, 875](#)
- [epptr, 876](#)
- [fd, 876](#)
- [file, 876](#)
- [gbump, 876](#)
- [getloc, 877](#)
- [gptr, 877](#)
- [imbue, 877](#)
- [in_avail, 878](#)
- [is_open, 878](#)
- [open, 878, 879](#)
- [overflow, 880](#)
- [pbackfail, 880](#)
- [pbase, 881](#)
- [pbump, 881](#)
- [pptr, 882](#)
- [pubimbue, 882](#)
- [pubseekoff, 882](#)
- [pubseekpos, 883](#)
- [pubsetbuf, 883](#)
- [pubsync, 883](#)
- [sbumpc, 883](#)
- [seekoff, 884](#)
- [seekpos, 884](#)
- [setbuf, 884](#)
- [setg, 885](#)
- [setp, 885](#)
- [sgetc, 886](#)
- [sgetn, 886](#)
- [showmanyc, 887](#)
- [snextc, 887](#)
- [sputbackc, 887](#)
- [sputc, 888](#)
- [sputn, 888](#)
- [stdio_filebuf, 873](#)
- [sungetc, 889](#)
- [sync, 889](#)
- [uflow, 889](#)
- [underflow, 890](#)
- [xsgetn, 890](#)
- [xsputn, 891](#)
- [__gnu_cxx::stdio_filebuf< _CharT, _Traits >, 870](#)
- [__gnu_cxx::stdio_sync_filebuf](#)
 - [_M_buf_locale, 913](#)
 - [_M_in_beg, 913](#)
 - [_M_in_cur, 913](#)
 - [_M_in_end, 913](#)
 - [_M_out_beg, 913](#)
 - [_M_out_cur, 913](#)
 - [_M_out_end, 914](#)
 - [eback, 898](#)
 - [egptr, 898](#)
 - [epptr, 899](#)
 - [file, 899](#)
 - [gbump, 899](#)
 - [getloc, 900](#)
 - [gptr, 900](#)
 - [imbue, 900](#)
 - [in_avail, 901](#)
 - [overflow, 901](#)
 - [pbackfail, 902](#)
 - [pbase, 902](#)
 - [pbump, 902](#)
 - [pptr, 903](#)
 - [pubimbue, 903](#)
 - [pubseekoff, 904](#)
 - [pubseekpos, 904](#)
 - [pubsetbuf, 904](#)
 - [pubsync, 904](#)
 - [sbumpc, 904](#)
 - [seekoff, 905](#)
 - [seekpos, 905](#)
 - [setbuf, 905](#)
 - [setg, 906](#)
 - [setp, 906](#)
 - [sgetc, 907](#)
 - [sgetn, 907](#)
 - [showmanyc, 907](#)
 - [snextc, 908](#)
 - [sputbackc, 908](#)
 - [sputc, 909](#)
 - [sputn, 909](#)
 - [sungetc, 910](#)
 - [sync, 910](#)
 - [uflow, 910](#)
 - [underflow, 911](#)
 - [xsgetn, 911](#)
 - [xsputn, 912](#)
- [__gnu_cxx::stdio_sync_filebuf< _CharT, _Traits >, 896](#)
- [__gnu_cxx::subtractive_rng, 914](#)
 - [argument_type, 915](#)
 - [operator\(\), 915](#)
 - [result_type, 915](#)
 - [subtractive_rng, 915](#)
- [__gnu_cxx::temporary_buffer](#)
 - [~temporary_buffer, 917](#)
 - [begin, 917](#)
 - [end, 917](#)
 - [requested_size, 918](#)
 - [size, 918](#)
 - [temporary_buffer, 917](#)

- `__gnu_cxx::temporary_buffer< _ForwardIterator, _Tp >`, 916
- `__gnu_cxx::throw_allocator_base< _Tp, _Cond >`, 918
- `__gnu_cxx::throw_allocator_limit< _Tp >`, 920
- `__gnu_cxx::throw_allocator_random< _Tp >`, 922
- `__gnu_cxx::throw_value_base< _Cond >`, 923
- `__gnu_cxx::throw_value_limit`, 925
- `__gnu_cxx::throw_value_random`, 926
- `__gnu_cxx::typelist`, 390
 - `apply_generator`, 390
- `__gnu_cxx::unary_compose`
 - `argument_type`, 928
 - `result_type`, 928
- `__gnu_cxx::unary_compose< _Operation1, _Operation2 >`, 927
- `__gnu_debug`, 390
 - `_Distance_precision`, 397
 - `_check_dereferenceable`, 397, 398
 - `_check_singular`, 398
 - `_check_singular_aux`, 398
 - `_check_string`, 398
 - `_foreign_iterator_aux2`, 398, 399
 - `_get_distance`, 399
 - `_valid_range`, 399, 400
 - `_valid_range_aux`, 400
- `__gnu_debug:: _After_nth_from< _Iterator >`, 929
- `__gnu_debug:: _BeforeBeginHelper< _Sequence >`, 929
- `__gnu_debug:: _Equal_to< _Type >`, 930
- `__gnu_debug:: _Not_equal_to< _Type >`, 930
- `__gnu_debug:: _Safe_container< _SafeContainer, _Alloc, _SafeBase, _IsCxx11AllocatorAware >`, 930
- `__gnu_debug:: _Safe_forward_list`
 - `_M_attach`, 932
 - `_M_attach_single`, 932
 - `_M_const_iterators`, 934
 - `_M_detach`, 932
 - `_M_detach_all`, 933
 - `_M_detach_single`, 933
 - `_M_detach_singular`, 933
 - `_M_get_mutex`, 933
 - `_M_invalidate_all`, 933
 - `_M_invalidate_if`, 933
 - `_M_iterators`, 934
 - `_M_revalidate_singular`, 933
 - `_M_transfer_from_if`, 933
 - `_M_version`, 934
- `__gnu_debug:: _Safe_forward_list< _SafeSequence >`, 931
- `__gnu_debug:: _Safe_iterator`
 - `_M_attach`, 938
 - `_M_attach_single`, 938
 - `_M_attached_to`, 938
 - `_M_before_dereferenceable`, 938
 - `_M_can_compare`, 938
- `_M_dereferenceable`, 939
- `_M_detach`, 939
- `_M_detach_single`, 939
- `_M_get_mutex`, 939
- `_M_incrementable`, 939
- `_M_invalidate`, 939
- `_M_is_before_begin`, 940
- `_M_is_begin`, 940
- `_M_is_beginnest`, 940
- `_M_is_end`, 940
- `_M_next`, 943
- `_M_prior`, 943
- `_M_reset`, 940
- `_M_sequence`, 944
- `_M_singular`, 940
- `_M_unlink`, 941
- `_M_version`, 944
- `_Safe_iterator`, 936, 937
- `base`, 941
- `operator _Iterator`, 941
- `operator*`, 941
- `operator++`, 941, 942
- `operator->`, 942
- `operator--`, 942
- `operator=`, 943
- `__gnu_debug:: _Safe_iterator< _Iterator, _Sequence >`, 934
- `__gnu_debug:: _Safe_iterator_base`, 944
 - `_M_attach`, 946
 - `_M_attach_single`, 946
 - `_M_attached_to`, 946
 - `_M_can_compare`, 946
 - `_M_detach`, 946
 - `_M_detach_single`, 947
 - `_M_get_mutex`, 947
 - `_M_invalidate`, 947
 - `_M_next`, 948
 - `_M_prior`, 948
 - `_M_reset`, 947
 - `_M_sequence`, 948
 - `_M_singular`, 947
 - `_M_unlink`, 947
 - `_M_version`, 948
 - `_Safe_iterator_base`, 945, 946
- `__gnu_debug:: _Safe_local_iterator`
 - `_M_attach`, 952
 - `_M_attach_single`, 952
 - `_M_attached_to`, 952
 - `_M_can_compare`, 953
 - `_M_dereferenceable`, 953
 - `_M_detach`, 953
 - `_M_detach_single`, 953
 - `_M_get_mutex`, 953
 - `_M_in_same_bucket`, 953

- [_M_incrementable, 954](#)
- [_M_invalidate, 954](#)
- [_M_is_begin, 954](#)
- [_M_is_end, 954](#)
- [_M_next, 958](#)
- [_M_prior, 958](#)
- [_M_reset, 954](#)
- [_M_sequence, 958](#)
- [_M_singular, 955](#)
- [_M_unlink, 955](#)
- [_M_version, 958](#)
- [_Safe_local_iterator, 951](#)
- [base, 955](#)
- [bucket, 955](#)
- [operator _iterator, 955](#)
- [operator*, 956](#)
- [operator++, 956](#)
- [operator->, 956](#)
- [operator=, 957](#)
- [__gnu_debug:: Safe_local_iterator< _iterator, Sequence >, 949](#)
- [__gnu_debug:: Safe_local_iterator_base, 959](#)
- [_M_attach, 961](#)
- [_M_attach_single, 961](#)
- [_M_attached_to, 961](#)
- [_M_can_compare, 961](#)
- [_M_detach, 961](#)
- [_M_detach_single, 961](#)
- [_M_get_mutex, 961](#)
- [_M_invalidate, 962](#)
- [_M_next, 962](#)
- [_M_prior, 962](#)
- [_M_reset, 962](#)
- [_M_sequence, 963](#)
- [_M_singular, 962](#)
- [_M_unlink, 962](#)
- [_M_version, 963](#)
- [_Safe_local_iterator_base, 960](#)
- [__gnu_debug:: Safe_node_sequence](#)
- [_M_attach, 964](#)
- [_M_attach_single, 964](#)
- [_M_const_iterators, 966](#)
- [_M_detach, 965](#)
- [_M_detach_all, 965](#)
- [_M_detach_single, 965](#)
- [_M_detach_singular, 965](#)
- [_M_get_mutex, 965](#)
- [_M_invalidate_all, 965](#)
- [_M_invalidate_if, 965](#)
- [_M_iterators, 966](#)
- [_M_revalidate_singular, 966](#)
- [_M_swap, 966](#)
- [_M_transfer_from_if, 966](#)
- [_M_version, 967](#)
- [__gnu_debug:: Safe_node_sequence< _Sequence >, 963](#)
- [__gnu_debug:: Safe_sequence](#)
- [_M_attach, 968](#)
- [_M_attach_single, 968](#)
- [_M_const_iterators, 970](#)
- [_M_detach, 968](#)
- [_M_detach_all, 968](#)
- [_M_detach_single, 969](#)
- [_M_detach_singular, 969](#)
- [_M_get_mutex, 969](#)
- [_M_invalidate_all, 969](#)
- [_M_invalidate_if, 969](#)
- [_M_iterators, 970](#)
- [_M_revalidate_singular, 969](#)
- [_M_swap, 969](#)
- [_M_transfer_from_if, 970](#)
- [_M_version, 970](#)
- [__gnu_debug:: Safe_sequence< _Sequence >, 967](#)
- [__gnu_debug:: Safe_sequence_base, 971](#)
- [_M_attach, 972](#)
- [_M_attach_single, 972](#)
- [_M_const_iterators, 974](#)
- [_M_detach, 972](#)
- [_M_detach_all, 973](#)
- [_M_detach_single, 973](#)
- [_M_detach_singular, 973](#)
- [_M_get_mutex, 973](#)
- [_M_invalidate_all, 973](#)
- [_M_iterators, 974](#)
- [_M_revalidate_singular, 973](#)
- [_M_swap, 973](#)
- [_M_version, 974](#)
- [~ Safe_sequence_base, 972](#)
- [__gnu_debug:: Safe_unordered_container](#)
- [_M_attach, 976](#)
- [_M_attach_local, 976](#)
- [_M_attach_local_single, 976](#)
- [_M_attach_single, 976](#)
- [_M_const_iterators, 978](#)
- [_M_const_local_iterators, 978](#)
- [_M_detach, 976](#)
- [_M_detach_all, 976](#)
- [_M_detach_local, 976](#)
- [_M_detach_local_single, 976](#)
- [_M_detach_single, 976](#)
- [_M_detach_singular, 976](#)
- [_M_get_mutex, 977](#)
- [_M_invalidate_all, 977](#)
- [_M_invalidate_if, 977](#)
- [_M_invalidate_local_if, 977](#)
- [_M_iterators, 978](#)
- [_M_local_iterators, 978](#)
- [_M_revalidate_singular, 977](#)

- `_M_swap`, 977, 978
 - `_M_version`, 978
- `__gnu_debug::Safe_unordered_container< _Container`
 - `>`, 974
- `__gnu_debug::Safe_unordered_container_base`, 979
 - `_M_attach`, 980
 - `_M_attach_local`, 980
 - `_M_attach_local_single`, 980
 - `_M_attach_single`, 980
 - `_M_const_iterators`, 982
 - `_M_const_local_iterators`, 982
 - `_M_detach`, 980
 - `_M_detach_all`, 981
 - `_M_detach_local`, 981
 - `_M_detach_local_single`, 981
 - `_M_detach_single`, 981
 - `_M_detach_singular`, 981
 - `_M_get_mutex`, 981
 - `_M_invalidate_all`, 981
 - `_M_iterators`, 982
 - `_M_local_iterators`, 982
 - `_M_revalidate_singular`, 981
 - `_M_swap`, 982
 - `_M_version`, 982
 - `~Safe_unordered_container_base`, 980
- `__gnu_debug::Safe_vector< _SafeSequence, _Base←
 Sequence >`, 983
- `__gnu_debug::Sequence_traits< _Sequence >`, 983
- `__gnu_debug::basic_string`
 - `_M_attach`, 989
 - `_M_attach_single`, 989
 - `_M_const_iterators`, 1009
 - `_M_detach`, 989
 - `_M_detach_all`, 989
 - `_M_detach_single`, 989
 - `_M_detach_singular`, 989
 - `_M_get_mutex`, 989
 - `_M_invalidate_all`, 990
 - `_M_invalidate_if`, 990
 - `_M_iterators`, 1009
 - `_M_revalidate_singular`, 990
 - `_M_swap`, 990
 - `_M_transfer_from_if`, 990
 - `_M_version`, 1009
- `append`, 990, 991
- `assign`, 991, 993
- `at`, 994
- `back`, 995
- `capacity`, 995
- `compare`, 995, 996
- `empty`, 996
- `erase`, 996, 997
- `find`, 997
- `find_first_not_of`, 997
- `find_first_of`, 998
- `find_last_not_of`, 998
- `find_last_of`, 999
- `front`, 999
- `get_allocator`, 999
- `insert`, 999–1002
- `length`, 1002
- `max_size`, 1002
- `npos`, 1009
- `operator+=`, 1002
- `replace`, 1003–1007
- `reserve`, 1007
- `rfind`, 1008
- `size`, 1008
- `swap`, 1008
- `__gnu_debug::basic_string< _CharT, _Traits, _Allocator`
 - `>`, 984
- `__gnu_internal`, 401
- `__gnu_parallel`, 401
 - `_AlgorithmStrategy`, 409
 - `_BinIndex`, 409
 - `_CASable`, 409
 - `_CASable_bits`, 450
 - `_CASable_mask`, 450
 - `_FindAlgorithm`, 409
 - `_MultiwayMergeAlgorithm`, 410
 - `_Parallelism`, 410
 - `_PartialSumAlgorithm`, 410
 - `_SequenceIndex`, 409
 - `_SortAlgorithm`, 410
 - `_SplittingAlgorithm`, 410
 - `_ThreadIndex`, 409
 - `__calc_borders`, 410
 - `__compare_and_swap`, 411
 - `__decode2`, 411
 - `__determine_samples`, 412
 - `__encode2`, 412
 - `__equally_split`, 412
 - `__equally_split_point`, 413
 - `__fetch_and_add`, 413
 - `__find_template`, 414, 415
 - `__for_each_template_random_access`, 416
 - `__for_each_template_random_access_ed`, 416
 - `__for_each_template_random_access_omp_loop`,
 417
 - `__for_each_template_random_access_omp_loop←
 _static`, 417
 - `__for_each_template_random_access_workstealing`,
 418
 - `__is_sorted`, 419
 - `__median_of_three_iterators`, 419
 - `__merge_advance`, 419
 - `__merge_advance_movc`, 420
 - `__merge_advance_usual`, 420

- `__parallel_merge_advance`, 421
- `__parallel_nth_element`, 422
- `__parallel_partial_sort`, 422
- `__parallel_partial_sum`, 424
- `__parallel_partial_sum_basecase`, 424
- `__parallel_partial_sum_linear`, 425
- `__parallel_partition`, 425
- `__parallel_random_shuffle`, 426
- `__parallel_random_shuffle_drs`, 426
- `__parallel_random_shuffle_drs_pu`, 427
- `__parallel_sort`, 427–431
- `__parallel_sort_qs`, 432
- `__parallel_sort_qs_conquer`, 432
- `__parallel_sort_qs_divide`, 433
- `__parallel_sort_qsb`, 433
- `__parallel_unique_copy`, 433, 435
- `__qsb_conquer`, 435
- `__qsb_divide`, 436
- `__qsb_local_sort_with_helping`, 436
- `__random_number_pow2`, 436
- `__rd_log2`, 437
- `__round_up_to_pow2`, 437
- `__search_template`, 437
- `__sequential_multiway_merge`, 438
- `__sequential_random_shuffle`, 438
- `__shrink`, 439
- `__shrink_and_double`, 439
- `__yield`, 440
- `list_partition`, 440
- `max`, 440
- `min`, 440
- `multiseq_partition`, 441
- `multiseq_selection`, 441
- `multiway_merge`, 442
- `multiway_merge_3_variant`, 443
- `multiway_merge_4_variant`, 444
- `multiway_merge_exact_splitting`, 445
- `multiway_merge_loser_tree`, 445
- `multiway_merge_loser_tree_sentinel`, 445
- `multiway_merge_loser_tree_unguarded`, 446
- `multiway_merge_sampling_splitting`, 447
- `multiway_merge_sentinels`, 447
- `parallel_balanced`, 410
- `parallel_multiway_merge`, 449
- `parallel_omp_loop`, 410
- `parallel_omp_loop_static`, 410
- `parallel_sort_mwms`, 449
- `parallel_sort_mwms_pu`, 450
- `parallel_taskqueue`, 410
- `parallel_unbalanced`, 410
- `sequential`, 410
- `__gnu_parallel::DRSSorterPU`
 - `_M_bins_begin`, 1051
 - `_M_num_threads`, 1051
- `__gnu_parallel::DRSSorterPU`
 - `_M_sd`, 1051
 - `_M_seed`, 1051
 - `__bins_end`, 1051
- `__gnu_parallel::DRSSorterPU`
 - `<_RAIter, _RandomGenerator>`, 1050
- `__gnu_parallel::DRandomShufflingGlobalData`
 - `_DRandomShufflingGlobalData`, 1049
 - `_M_bin_proc`, 1049
 - `_M_dist`, 1049
 - `_M_num_bins`, 1049
 - `_M_num_bits`, 1049
 - `_M_source`, 1049
 - `_M_starts`, 1050
 - `_M_temporaries`, 1050
- `__gnu_parallel::DRandomShufflingGlobalData`
 - `<_RAIter>`, 1048
- `__gnu_parallel::DummyReduct`, 1052
- `__gnu_parallel::EqualFromLess`
 - `first_argument_type`, 1053
 - `result_type`, 1053
 - `second_argument_type`, 1053
- `__gnu_parallel::EqualFromLess`
 - `<_T1, _T2, _Compare>`, 1052
- `__gnu_parallel::EqualTo`
 - `first_argument_type`, 1054
 - `result_type`, 1054
 - `second_argument_type`, 1054
- `__gnu_parallel::EqualTo`
 - `<_T1, _T2>`, 1053
- `__gnu_parallel::GuardedIterator`
 - `_GuardedIterator`, 1055
 - `operator _RAIter`, 1056
 - `operator<`, 1056
 - `operator<=`, 1057
 - `operator*`, 1056
 - `operator++`, 1056
- `__gnu_parallel::GuardedIterator`
 - `<_RAIter, _Compare>`, 1055
- `__gnu_parallel::IteratorPair`
 - `_PCCP`, 1058
 - `_PCCFP`, 1058
 - `first`, 1059
 - `second`, 1059
 - `second_type`, 1059
- `__gnu_parallel::IteratorPair`
 - `<_Iterator1, _Iterator2, _IteratorCategory>`, 1057
- `__gnu_parallel::IteratorTriple`
 - `<_Iterator1, _Iterator2, _Iterator3, _IteratorCategory>`, 1059
- `__gnu_parallel::Job`
 - `_M_first`, 1061
 - `_M_last`, 1061
 - `_M_load`, 1061
- `__gnu_parallel::Job`
 - `<_DifferenceTp>`, 1060
- `__gnu_parallel::Less`
 - `first_argument_type`, 1062

result_type, 1062
 second_argument_type, 1063
 __gnu_parallel::Less< _T1, _T2 >, 1062
 __gnu_parallel::Lexicographic
 first_argument_type, 1064
 result_type, 1064
 second_argument_type, 1064
 __gnu_parallel::Lexicographic< _T1, _T2, _Compare >, 1063
 __gnu_parallel::LexicographicReverse
 first_argument_type, 1065
 result_type, 1065
 second_argument_type, 1065
 __gnu_parallel::LexicographicReverse< _T1, _T2, _Compare >, 1064
 __gnu_parallel::LoserTree
 _M_comp, 1067
 _M_first_insert, 1068
 _M_log_k, 1068
 _M_losers, 1068
 __delete_min_insert, 1067
 __get_min_source, 1067
 __insert_start, 1067
 __gnu_parallel::LoserTree< __stable, _Tp, _Compare >, 1066
 __gnu_parallel::LoserTree< false, _Tp, _Compare >, 1068
 _M_comp, 1071
 _M_first_insert, 1071
 _M_log_k, 1071
 _M_losers, 1071
 __delete_min_insert, 1069
 __get_min_source, 1070
 __init_winner, 1070
 __insert_start, 1070
 __gnu_parallel::LoserTreeBase
 LoserTreeBase, 1073
 _M_comp, 1074
 _M_first_insert, 1074
 _M_log_k, 1074
 _M_losers, 1074
 __get_min_source, 1073
 __insert_start, 1073
 ~LoserTreeBase, 1073
 __gnu_parallel::LoserTreeBase< _Tp, _Compare >, 1072
 __gnu_parallel::LoserTreeBase< _Tp, _Compare >::__Loser, 1075
 __gnu_parallel::LoserTreeBase::__Loser
 _M_key, 1075
 _M_source, 1075
 _M_sup, 1076
 __gnu_parallel::LoserTreePointer< __stable, _Tp, _Compare >, 1076
 __gnu_parallel::LoserTreePointer< false, _Tp, _Compare >, 1077
 __gnu_parallel::LoserTreePointerBase< _Tp, _Compare >, 1078
 __gnu_parallel::LoserTreePointerBase< _Tp, _Compare >::__Loser, 1079
 __gnu_parallel::LoserTreePointerUnguarded< __stable, _Tp, _Compare >, 1080
 __gnu_parallel::LoserTreePointerUnguarded< false, _Tp, _Compare >, 1081
 __gnu_parallel::LoserTreePointerUnguardedBase< _Tp, _Compare >, 1082
 __gnu_parallel::LoserTreeTraits
 _M_use_pointer, 1084
 __gnu_parallel::LoserTreeTraits< _Tp >, 1083
 __gnu_parallel::LoserTreeUnguarded< __stable, _Tp, _Compare >, 1084
 __gnu_parallel::LoserTreeUnguarded< false, _Tp, _Compare >, 1085
 __gnu_parallel::LoserTreeUnguardedBase< _Tp, _Compare >, 1086
 __gnu_parallel::Multiplies
 first_argument_type, 1088
 result_type, 1088
 second_argument_type, 1088
 __gnu_parallel::Multiplies< _Tp1, _Tp2, _Result >, 1087
 __gnu_parallel::Nothing, 1089
 operator(), 1089
 __gnu_parallel::PMWMSSortingData
 _M_num_threads, 1092
 _M_offsets, 1092
 _M_pieces, 1092
 _M_samples, 1093
 _M_source, 1093
 _M_starts, 1093
 _M_temporary, 1093
 __gnu_parallel::PMWMSSortingData< _RAIter >, 1092
 __gnu_parallel::Piece
 _M_begin, 1090
 _M_end, 1090
 __gnu_parallel::Piece< _DifferenceTp >, 1089
 __gnu_parallel::Plus
 first_argument_type, 1091
 result_type, 1091
 second_argument_type, 1091
 __gnu_parallel::Plus< _Tp1, _Tp2, _Result >, 1090
 __gnu_parallel::PseudoSequence
 _PseudoSequence, 1094
 begin, 1094
 end, 1095
 __gnu_parallel::PseudoSequence< _Tp, _DifferenceTp >, 1094
 __gnu_parallel::PseudoSequenceIterator< _Tp, _DifferenceTp >, 1095

- __gnu_parallel:: QSBThreadLocal
 - _M_elements_leftover, 1097
 - _M_global, 1097
 - _M_initial, 1097
 - _M_leftover_parts, 1097
 - _M_num_threads, 1097
 - _Piece, 1096
 - _QSBThreadLocal, 1096
- __gnu_parallel:: QSBThreadLocal< _RAIter >, 1096
- __gnu_parallel:: RandomNumber, 1098
 - _RandomNumber, 1098
 - _genrand_bits, 1098
 - operator(), 1099
- __gnu_parallel:: RestrictedBoundedConcurrentQueue
 - _RestrictedBoundedConcurrentQueue, 1100
 - ~_RestrictedBoundedConcurrentQueue, 1100
 - pop_back, 1100
 - pop_front, 1100
 - push_front, 1100
- __gnu_parallel:: RestrictedBoundedConcurrentQueue<
 - _Tp >, 1099
- __gnu_parallel:: SamplingSorter< __stable, _RAIter, _<
 - StrictWeakOrdering >, 1101
- __gnu_parallel:: SamplingSorter< false, _RAIter, _<
 - StrictWeakOrdering >, 1101
- __gnu_parallel:: Settings, 1102
 - accumulate_minimal_n, 1103
 - adjacent_difference_minimal_n, 1103
 - cache_line_size, 1104
 - count_minimal_n, 1104
 - fill_minimal_n, 1104
 - find_increasing_factor, 1104
 - find_initial_block_size, 1104
 - find_maximum_block_size, 1104
 - find_scale_factor, 1104
 - find_sequential_search_size, 1105
 - for_each_minimal_n, 1105
 - generate_minimal_n, 1105
 - get, 1103
 - L1_cache_size, 1105
 - L2_cache_size, 1105
 - max_element_minimal_n, 1105
 - merge_minimal_n, 1105
 - merge_oversampling, 1106
 - min_element_minimal_n, 1106
 - multiway_merge_minimal_k, 1106
 - multiway_merge_minimal_n, 1106
 - multiway_merge_oversampling, 1106
 - nth_element_minimal_n, 1106
 - partial_sort_minimal_n, 1106
 - partial_sum_dilation, 1106
 - partial_sum_minimal_n, 1107
 - partition_chunk_share, 1107
 - partition_chunk_size, 1107
 - partition_minimal_n, 1107
 - qsb_steals, 1107
 - random_shuffle_minimal_n, 1107
 - replace_minimal_n, 1107
 - search_minimal_n, 1108
 - set, 1103
 - set_difference_minimal_n, 1108
 - set_intersection_minimal_n, 1108
 - set_symmetric_difference_minimal_n, 1108
 - set_union_minimal_n, 1108
 - sort_minimal_n, 1108
 - sort_mwms_oversampling, 1108
 - sort_qs_num_samples_preset, 1108
 - sort_qsb_base_case_maximal_n, 1109
 - TLB_size, 1109
 - transform_minimal_n, 1109
 - unique_copy_minimal_n, 1109
- __gnu_parallel:: SplitConsistently< __exact, _RAIter, _<
 - Compare, _SortingPlacesIterator >, 1109
- __gnu_parallel:: SplitConsistently< false, _RAIter, _<
 - Compare, _SortingPlacesIterator >, 1110
- __gnu_parallel:: SplitConsistently< true, _RAIter, _<
 - Compare, _SortingPlacesIterator >, 1110
- __gnu_parallel:: __accumulate_binop_reduct< _BinOp >,
 - 1009
- __gnu_parallel:: __accumulate_selector
 - _M_finish_iterator, 1011
 - operator(), 1011
- __gnu_parallel:: __accumulate_selector< _It >, 1010
- __gnu_parallel:: __adjacent_difference_selector
 - _M_finish_iterator, 1012
- __gnu_parallel:: __adjacent_difference_selector< _It >,
 - 1011
- __gnu_parallel:: __adjacent_find_selector, 1012
 - _M_sequential_algorithm, 1013
 - operator(), 1013
- __gnu_parallel:: __binder1st
 - argument_type, 1015
 - result_type, 1015
- __gnu_parallel:: __binder1st< _Operation, _First<
 - ArgumentType, _SecondArgumentType, _<
 - ResultType >, 1014
- __gnu_parallel:: __binder2nd
 - argument_type, 1016
 - result_type, 1016
- __gnu_parallel:: __binder2nd< _Operation, _First<
 - ArgumentType, _SecondArgumentType, _<
 - ResultType >, 1015
- __gnu_parallel:: __count_if_selector
 - _M_finish_iterator, 1018
 - operator(), 1017
- __gnu_parallel:: __count_if_selector< _It, _Diff >, 1017
- __gnu_parallel:: __count_selector
 - _M_finish_iterator, 1019

operator(), 1019
 __gnu_parallel::__count_selector< _It, _Diff >, 1018
 __gnu_parallel::__fill_selector
 _M_finish_iterator, 1021
 operator(), 1020
 __gnu_parallel::__fill_selector< _It >, 1020
 __gnu_parallel::__find_first_of_selector
 _M_sequential_algorithm, 1022
 operator(), 1022
 __gnu_parallel::__find_first_of_selector< _FIterator >, 1021
 __gnu_parallel::__find_if_selector, 1023
 _M_sequential_algorithm, 1023
 operator(), 1024
 __gnu_parallel::__for_each_selector
 _M_finish_iterator, 1025
 operator(), 1025
 __gnu_parallel::__for_each_selector< _It >, 1024
 __gnu_parallel::__generate_selector
 _M_finish_iterator, 1027
 operator(), 1026
 __gnu_parallel::__generate_selector< _It >, 1026
 __gnu_parallel::__generic_find_selector, 1027
 __gnu_parallel::__generic_for_each_selector
 _M_finish_iterator, 1030
 __gnu_parallel::__generic_for_each_selector< _It >, 1029
 __gnu_parallel::__identity_selector
 _M_finish_iterator, 1031
 operator(), 1031
 __gnu_parallel::__identity_selector< _It >, 1030
 __gnu_parallel::__inner_product_selector
 _M_finish_iterator, 1033
 __begin1_iterator, 1033
 __begin2_iterator, 1033
 __inner_product_selector, 1032
 operator(), 1033
 __gnu_parallel::__inner_product_selector< _It, _It2, _Tp >, 1032
 __gnu_parallel::__max_element_reduct< _Compare, _It >, 1034
 __gnu_parallel::__min_element_reduct< _Compare, _It >, 1034
 __gnu_parallel::__mismatch_selector, 1035
 _M_sequential_algorithm, 1036
 operator(), 1036
 __gnu_parallel::__multiway_merge_3_variant_sentinel↵
 switch< __sentinels, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >, 1036
 __gnu_parallel::__multiway_merge_3_variant_sentinel↵
 switch< true, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >, 1037
 __gnu_parallel::__multiway_merge_4_variant_sentinel↵
 switch< __sentinels, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >, 1037
 __gnu_parallel::__multiway_merge_4_variant_sentinel↵
 switch< true, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >, 1038
 __gnu_parallel::__multiway_merge_k_variant_sentinel↵
 switch< __sentinels, __stable, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >, 1038
 __gnu_parallel::__multiway_merge_k_variant_sentinel↵
 switch< false, __stable, _RAIterIterator, _RAIter3, _DifferenceTp, _Compare >, 1039
 __gnu_parallel::__replace_if_selector
 _M_finish_iterator, 1041
 __new_val, 1041
 __replace_if_selector, 1040
 operator(), 1041
 __gnu_parallel::__replace_if_selector< _It, _Op, _Tp >, 1040
 __gnu_parallel::__replace_selector
 _M_finish_iterator, 1043
 __new_val, 1043
 __replace_selector, 1042
 operator(), 1043
 __gnu_parallel::__replace_selector< _It, _Tp >, 1042
 __gnu_parallel::__transform1_selector
 _M_finish_iterator, 1045
 operator(), 1044
 __gnu_parallel::__transform1_selector< _It >, 1044
 __gnu_parallel::__transform2_selector
 _M_finish_iterator, 1046
 operator(), 1046
 __gnu_parallel::__transform2_selector< _It >, 1045
 __gnu_parallel::__unary_negate
 argument_type, 1048
 result_type, 1048
 __gnu_parallel::__unary_negate< _Predicate, argument_type >, 1047
 __gnu_parallel::balanced_quicksort_tag, 1111
 __get_num_threads, 1111
 set_num_threads, 1111
 __gnu_parallel::balanced_tag, 1112
 __get_num_threads, 1112
 set_num_threads, 1112
 __gnu_parallel::constant_size_blocks_tag, 1113
 __gnu_parallel::default_parallel_tag, 1114
 __get_num_threads, 1114
 set_num_threads, 1114
 __gnu_parallel::equal_split_tag, 1115
 __gnu_parallel::exact_tag, 1116
 __get_num_threads, 1116
 set_num_threads, 1116
 __gnu_parallel::find_tag, 1117
 __gnu_parallel::growing_blocks_tag, 1118
 __gnu_parallel::multiway_mergesort_exact_tag, 1118
 __get_num_threads, 1119

- set_num_threads, 1119
- __gnu_parallel::multiway_mergesort_sampling_tag, 1120
 - __get_num_threads, 1120
 - set_num_threads, 1120
- __gnu_parallel::multiway_mergesort_tag, 1121
 - __get_num_threads, 1122
 - set_num_threads, 1122
- __gnu_parallel::omp_loop_static_tag, 1122
 - __get_num_threads, 1123
 - set_num_threads, 1123
- __gnu_parallel::omp_loop_tag, 1124
 - __get_num_threads, 1124
 - set_num_threads, 1124
- __gnu_parallel::parallel_tag, 1126
 - __get_num_threads, 1127
 - parallel_tag, 1127
 - set_num_threads, 1127
- __gnu_parallel::quicksort_tag, 1128
 - __get_num_threads, 1129
 - set_num_threads, 1129
- __gnu_parallel::sampling_tag, 1129
 - __get_num_threads, 1130
 - set_num_threads, 1130
- __gnu_parallel::sequential_tag, 1130
- __gnu_parallel::unbalanced_tag, 1131
 - __get_num_threads, 1131
 - set_num_threads, 1131
- __gnu_pbds, 451
- __gnu_pbds::associative_tag, 1132
- __gnu_pbds::basic_branch< Key, Mapped, Tag, Node_↵
Update, Policy_Tl, _Alloc >, 1132
- __gnu_pbds::basic_branch_tag, 1134
- __gnu_pbds::basic_hash_table< Key, Mapped, Hash_↵
_Fn, Eq_Fn, Resize_Policy, Store_Hash, Tag,
Policy_Tl, _Alloc >, 1134
- __gnu_pbds::basic_hash_tag, 1136
- __gnu_pbds::basic_invalidation_guarantee, 1137
- __gnu_pbds::binary_heap_tag, 1138
- __gnu_pbds::binomial_heap_tag, 1139
- __gnu_pbds::cc_hash_max_collision_check_resize_↵
trigger
 - cc_hash_max_collision_check_resize_trigger, 1140
 - external_load_access, 1140
 - get_load, 1141
 - is_grow_needed, 1141
 - is_resize_needed, 1141
 - notify_cleared, 1141
 - notify_erase_search_collision, 1141
 - notify_erase_search_end, 1141
 - notify_erase_search_start, 1141
 - notify_erased, 1142
 - notify_externally_resized, 1142
 - notify_find_search_collision, 1142
 - notify_find_search_end, 1142
 - notify_find_search_start, 1142
 - notify_insert_search_collision, 1142
 - notify_insert_search_end, 1143
 - notify_insert_search_start, 1143
 - notify_inserted, 1143
 - notify_resized, 1143
 - set_load, 1143
- __gnu_pbds::cc_hash_max_collision_check_resize_↵
trigger< External_Load_Access, Size_Type >, 1139
- __gnu_pbds::cc_hash_table
 - cc_hash_table, 1145–1147
- __gnu_pbds::cc_hash_table< Key, Mapped, Hash_↵
Fn, Eq_Fn, Comb_Hash_Fn, Resize_Policy,
Store_Hash, _Alloc >, 1144
- __gnu_pbds::cc_hash_tag, 1148
- __gnu_pbds::container_error, 1149
 - what, 1149
- __gnu_pbds::container_tag, 1150
- __gnu_pbds::container_traits
 - erase_can_throw, 1151
 - order_preserving, 1151
 - reverse_iteration, 1151
 - split_join_can_throw, 1151
- __gnu_pbds::container_traits< Cntnr >, 1150
- __gnu_pbds::container_traits_base< _Tag >, 1151
- __gnu_pbds::container_traits_base< binary_heap_tag >, 1152
- __gnu_pbds::container_traits_base< binomial_heap_tag
>, 1152
- __gnu_pbds::container_traits_base< cc_hash_tag >, 1153
- __gnu_pbds::container_traits_base< gp_hash_tag >, 1153
- __gnu_pbds::container_traits_base< list_update_tag >, 1154
- __gnu_pbds::container_traits_base< ov_tree_tag >, 1154
- __gnu_pbds::container_traits_base< pairing_heap_tag
>, 1155
- __gnu_pbds::container_traits_base< pat_trie_tag >, 1155
- __gnu_pbds::container_traits_base< rb_tree_tag >, 1156
- __gnu_pbds::container_traits_base< rc_binomial_heap_↵
_tag >, 1156
- __gnu_pbds::container_traits_base< splay_tree_tag >, 1157
- __gnu_pbds::container_traits_base< thin_heap_tag >, 1157
- __gnu_pbds::detail::bin_search_tree_const_it_↵< Node_↵
_Pointer, Value_Type, Pointer, Const_Pointer,
Reference, Const_Reference, Is_Forward_↵
Iterator, _Alloc >, 1158
- __gnu_pbds::detail::bin_search_tree_const_node_it_↵

- const_reference, 1160
- difference_type, 1160
- get_l_child, 1162
- get_metadata, 1162
- get_r_child, 1162
- iterator_category, 1161
- metadata_const_reference, 1161
- metadata_type, 1161
- operator!=, 1162
- operator*, 1162
- operator==, 1162
- reference, 1161
- value_type, 1161
- __gnu_pbds::detail::bin_search_tree_const_node_it< Node, Const_Iterator, Iterator, _Alloc >, 1159
- __gnu_pbds::detail::bin_search_tree_it< Node_Pointer, Value_Type, Pointer, Const_Pointer, Reference, Const_Reference, Is_Forward_Iterator, _Alloc >, 1163
- __gnu_pbds::detail::bin_search_tree_node_it< const_reference, 1166
- difference_type, 1166
- get_l_child, 1167
- get_metadata, 1167
- get_r_child, 1167
- iterator_category, 1166
- metadata_const_reference, 1166
- metadata_type, 1166
- operator!=, 1167
- operator*, 1167
- operator==, 1167
- reference, 1166
- value_type, 1166
- __gnu_pbds::detail::bin_search_tree_node_it< Node, Const_Iterator, Iterator, _Alloc >, 1165
- __gnu_pbds::detail::bin_search_tree_traits< node_const_iterator, 1169
- __gnu_pbds::detail::bin_search_tree_traits< Key, Mapped, Cmp_Fn, Node_Update, Node, _Alloc >, 1168
- __gnu_pbds::detail::bin_search_tree_traits< Key, null_< type, Cmp_Fn, Node_Update, Node, _Alloc >, 1169
- node_const_iterator, 1170
- __gnu_pbds::detail::binary_heap< Value_Type, Cmp_Fn, _Alloc >, 1170
- __gnu_pbds::detail::binary_heap_const_iterator< binary_heap_const_iterator_, 1174
- const_pointer, 1173
- const_reference, 1173
- difference_type, 1174
- iterator_category, 1174
- operator!=, 1175
- operator*, 1175
- operator->, 1175
- operator==, 1175
- pointer, 1174
- reference, 1174
- value_type, 1174
- __gnu_pbds::detail::binary_heap_const_iterator_< Value_Type, Entry, Simple, _Alloc >, 1172
- __gnu_pbds::detail::binary_heap_point_const_iterator< binary_heap_point_const_iterator_, 1178
- const_pointer, 1177
- const_reference, 1177
- difference_type, 1177
- iterator_category, 1177
- operator!=, 1179
- operator*, 1179
- operator->, 1179
- operator==, 1179
- pointer, 1178
- reference, 1178
- value_type, 1178
- __gnu_pbds::detail::binary_heap_point_const_iterator_< Value_Type, Entry, Simple, _Alloc >, 1176
- __gnu_pbds::detail::binomial_heap< Value_Type, Cmp_< Fn, _Alloc >, 1179
- __gnu_pbds::detail::binomial_heap_base< Value_Type, Cmp_Fn, _Alloc >, 1182
- __gnu_pbds::detail::branch_policy< Node_Cltr, Node_< Cltr, _Alloc >, 1185
- __gnu_pbds::detail::branch_policy< Node_Cltr, Node_Itr, _Alloc >, 1184
- __gnu_pbds::detail::cc_ht_map< empty, 1189
- get_comb_hash_fn, 1189
- get_eq_fn, 1189
- get_hash_fn, 1189, 1190
- get_resize_policy, 1190
- __gnu_pbds::detail::cc_ht_map< Key, Mapped, Hash_< Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Hash_< Fn, Resize_Policy >, 1186
- __gnu_pbds::detail::cond_dealtor< Entry, _Alloc >, 1190
- __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, Tag, Policy_TI >, 1191
- __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, cc_hash_tag, Policy_TI >, 1195
- type, 1195
- __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, gp_hash_tag, Policy_TI >, 1196
- type, 1196
- __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, list_update_tag, Policy_TI >, 1196
- type, 1197
- __gnu_pbds::detail::container_base_dispatch< Key,

- Mapped, _Alloc, ov_tree_tag, Policy_TI >, [1197](#)
- type, [1197](#)
- __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, pat_trie_tag, Policy_TI >, [1198](#)
- __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, rb_tree_tag, Policy_TI >, [1198](#)
- type, [1198](#)
- __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, splay_tree_tag, Policy_TI >, [1199](#)
- type, [1199](#)
- __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, cc_hash_tag, Policy_TI >, [1199](#)
- type, [1200](#)
- __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, gp_hash_tag, Policy_TI >, [1200](#)
- type, [1200](#)
- __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, list_update_tag, Policy_TI >, [1201](#)
- type, [1201](#)
- __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, ov_tree_tag, Policy_TI >, [1201](#)
- type, [1202](#)
- __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, pat_trie_tag, Policy_TI >, [1202](#)
- type, [1202](#)
- __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, rb_tree_tag, Policy_TI >, [1203](#)
- __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, splay_tree_tag, Policy_TI >, [1203](#)
- type, [1203](#)
- __gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, binary_heap_tag, null_type >, [1192](#)
- type, [1192](#)
- __gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, binomial_heap_tag, null_type >, [1192](#)
- type, [1193](#)
- __gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, pairing_heap_tag, null_type >, [1193](#)
- type, [1193](#)
- __gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, rc_binomial_heap_tag, null_↵ type >, [1194](#)
- type, [1194](#)
- __gnu_pbds::detail::container_base_dispatch< _VTp, Cmp_Fn, _Alloc, thin_heap_tag, null_type >, [1194](#)
- type, [1195](#)
- __gnu_pbds::detail::default_comb_hash_fn, [1204](#)
- type, [1204](#)
- __gnu_pbds::detail::default_eq_fn
- type, [1205](#)
- __gnu_pbds::detail::default_eq_fn< Key >, [1204](#)
- __gnu_pbds::detail::default_hash_fn
- type, [1205](#)
- __gnu_pbds::detail::default_hash_fn< Key >, [1205](#)
- __gnu_pbds::detail::default_probe_fn
- type, [1206](#)
- __gnu_pbds::detail::default_probe_fn< Comb_Probe_Fn >, [1205](#)
- __gnu_pbds::detail::default_resize_policy
- type, [1206](#)
- __gnu_pbds::detail::default_resize_policy< Comb_↵ Hash_Fn >, [1206](#)
- __gnu_pbds::detail::default_trie_access_traits< Key >, [1207](#)
- __gnu_pbds::detail::default_trie_access_traits< std_↵ ::basic_string< Char, Char_Traits, std_↵ ::allocator< char > > >, [1207](#)
- type, [1207](#)
- __gnu_pbds::detail::default_update_policy, [1208](#)
- type, [1208](#)
- __gnu_pbds::detail::dumnode_const_iterator< Key, Data, _Alloc >, [1208](#)
- __gnu_pbds::detail::entry_cmp< _VTp, Cmp_Fn, _Alloc, No_Throw >, [1209](#)
- __gnu_pbds::detail::entry_cmp< _VTp, Cmp_Fn, _Alloc, false >, [1209](#)
- __gnu_pbds::detail::entry_cmp< _VTp, Cmp_Fn, _Alloc, false >::type, [1209](#)
- __gnu_pbds::detail::entry_cmp< _VTp, Cmp_Fn, _Alloc, true >, [1210](#)
- type, [1210](#)
- __gnu_pbds::detail::entry_pred< _VTp, Pred, _Alloc, No_Throw >, [1211](#)
- __gnu_pbds::detail::entry_pred< _VTp, Pred, _Alloc, false >, [1211](#)
- __gnu_pbds::detail::entry_pred< _VTp, Pred, _Alloc, true >, [1211](#)
- __gnu_pbds::detail::eq_by_less< Key, Cmp_Fn >, [1212](#)
- __gnu_pbds::detail::gp_ht_map
- empty, [1215](#)
- get_comb_probe_fn, [1215](#)
- get_eq_fn, [1216](#)
- get_hash_fn, [1216](#)
- get_probe_fn, [1216](#)
- get_resize_policy, [1217](#)
- __gnu_pbds::detail::gp_ht_map< Key, Mapped, Hash_↵

Fn, Eq_Fn, _Alloc, Store_Hash, Comb_Probe↔
 _Fn, Probe_Fn, Resize_Policy >, [1212](#)
 __gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc,
 Store_Hash >, [1217](#)
 __gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc,
 false >, [1218](#)
 __gnu_pbds::detail::hash_eq_fn< Key, Eq_Fn, _Alloc,
 true >, [1219](#)
 __gnu_pbds::detail::hash_load_check_resize_trigger↔
 size_base< Size_Type, Hold_Size >, [1219](#)
 __gnu_pbds::detail::hash_load_check_resize_trigger↔
 size_base< Size_Type, true >, [1220](#)
 __gnu_pbds::detail::left_child_next_sibling_heap< Value↔
 _Type, Cmp_Fn, Node_Metadata, _Alloc >,
[1220](#)
 __gnu_pbds::detail::left_child_next_sibling_heap_const↔
 iterator
 const_pointer, [1223](#)
 const_reference, [1223](#)
 difference_type, [1223](#)
 iterator_category, [1223](#)
 left_child_next_sibling_heap_const_iterator_, [1224](#)
 operator!=, [1224](#)
 operator*, [1224](#)
 operator->, [1225](#)
 operator==, [1225](#)
 pointer, [1223](#)
 reference, [1223](#)
 value_type, [1224](#)
 __gnu_pbds::detail::left_child_next_sibling_heap_const↔
 iterator< Node, _Alloc >, [1222](#)
 __gnu_pbds::detail::left_child_next_sibling_heap_node↔
 _< _Value, _Metadata, _Alloc >, [1225](#)
 __gnu_pbds::detail::left_child_next_sibling_heap_node↔
 _point_const_iterator_
 const_pointer, [1227](#)
 const_reference, [1227](#)
 difference_type, [1228](#)
 iterator_category, [1228](#)
 left_child_next_sibling_heap_node_point_const↔
 iterator_, [1229](#)
 operator!=, [1229](#)
 operator*, [1229](#)
 operator->, [1229](#)
 operator==, [1229](#)
 pointer, [1228](#)
 reference, [1228](#)
 value_type, [1228](#)
 __gnu_pbds::detail::left_child_next_sibling_heap_node↔
 _point_const_iterator_< Node, _Alloc >, [1226](#)
 __gnu_pbds::detail::lu_counter_metadata< Size_Type >,
[1230](#)
 __gnu_pbds::detail::lu_counter_policy_base< Size_Type
 >, [1230](#)
 __gnu_pbds::detail::lu_map< Key, Mapped, Eq_Fn, _↔
 Alloc, Update_Policy >, [1231](#)
 __gnu_pbds::detail::mask_based_range_hashing<
 Size_Type >, [1233](#)
 __gnu_pbds::detail::mod_based_range_hashing< Size↔
 _Type >, [1234](#)
 __gnu_pbds::detail::no_throw_copies< Key, Mapped >,
[1235](#)
 __gnu_pbds::detail::no_throw_copies< Key, null_type >,
[1236](#)
 __gnu_pbds::detail::ov_tree_map
 node_begin, [1238](#)
 node_end, [1239](#)
 __gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp↔
 _Fn, Node_And_It_Traits, _Alloc >, [1236](#)
 __gnu_pbds::detail::ov_tree_map< Key, Mapped, Cmp↔
 _Fn, Node_And_It_Traits, _Alloc >::cond_dtor<
 Size_Type >, [1239](#)
 __gnu_pbds::detail::ov_tree_node_const_it_
 get_l_child, [1241](#)
 get_r_child, [1241](#)
 __gnu_pbds::detail::ov_tree_node_const_it_< Value↔
 Type, Metadata_Type, _Alloc >, [1240](#)
 __gnu_pbds::detail::ov_tree_node_it_
 get_l_child, [1243](#)
 get_r_child, [1243](#)
 operator*, [1243](#)
 __gnu_pbds::detail::ov_tree_node_it_< Value_Type,
 Metadata_Type, _Alloc >, [1242](#)
 __gnu_pbds::detail::pairing_heap< Value_Type, Cmp_Fn,
 _Alloc >, [1244](#)
 __gnu_pbds::detail::pat_trie_base, [1246](#)
 node_type, [1247](#)
 __gnu_pbds::detail::pat_trie_base::CIter< Node, Leaf,
 Head, Inode, Is_Forward_Iterator >, [1247](#)
 __gnu_pbds::detail::pat_trie_base::Head< _ATraits,
 Metadata >, [1249](#)
 __gnu_pbds::detail::pat_trie_base::Inode< _ATraits,
 Metadata >, [1250](#)
 __gnu_pbds::detail::pat_trie_base::Inode< _ATraits,
 Metadata >::const_iterator, [1252](#)
 __gnu_pbds::detail::pat_trie_base::Inode< _ATraits,
 Metadata >::iterator, [1253](#)
 __gnu_pbds::detail::pat_trie_base::Iter< Node, Leaf,
 Head, Inode, Is_Forward_Iterator >, [1255](#)
 __gnu_pbds::detail::pat_trie_base::Leaf< _ATraits,
 Metadata >, [1257](#)
 __gnu_pbds::detail::pat_trie_base::Metadata< Meta-
 data, _Alloc >, [1258](#)
 __gnu_pbds::detail::pat_trie_base::Metadata< null↔
 type, _Alloc >, [1259](#)
 __gnu_pbds::detail::pat_trie_base::Node_base< _A↔
 Traits, Metadata >, [1259](#)
 __gnu_pbds::detail::pat_trie_base::Node_citer

- `__rebind_m`, [1262](#)
- `get_child`, [1262](#)
- `get_metadata`, [1262](#)
- `metadata_type`, [1262](#)
- `num_children`, [1262](#)
- `operator!=`, [1262](#)
- `operator*`, [1263](#)
- `operator==`, [1263](#)
- `valid_prefix`, [1263](#)
- `__gnu_pbds::detail::pat_trie_base::_Node_citer` `< Node, Leaf, Head, Inode, _Cliterator, Iterator, _Alloc >`, [1260](#)
- `__gnu_pbds::detail::pat_trie_base::_Node_iter`
 - `__rebind_m`, [1265](#)
 - `get_child`, [1265](#)
 - `get_metadata`, [1265](#)
 - `metadata_type`, [1265](#)
 - `num_children`, [1265](#)
 - `operator!=`, [1265](#)
 - `operator*`, [1266](#)
 - `operator==`, [1266](#)
 - `valid_prefix`, [1266](#)
- `__gnu_pbds::detail::pat_trie_base::_Node_iter` `< Node, Leaf, Head, Inode, _Cliterator, Iterator, _Alloc >`, [1263](#)
- `__gnu_pbds::detail::pat_trie_map`
 - `node_begin`, [1269](#)
 - `node_end`, [1269](#)
 - `node_type`, [1269](#)
- `__gnu_pbds::detail::pat_trie_map` `< Key, Mapped, Node_↵_And_It_Traits, _Alloc >`, [1266](#)
- `__gnu_pbds::detail::probe_fn_base` `< _Alloc >`, [1270](#)
- `__gnu_pbds::detail::ranged_hash_fn` `< Key, Hash_Fn, _↵_Alloc, Comb_Hash_Fn, Store_Hash >`, [1270](#)
- `__gnu_pbds::detail::ranged_hash_fn` `< Key, Hash_Fn, _↵_Alloc, Comb_Hash_Fn, false >`, [1271](#)
- `__gnu_pbds::detail::ranged_hash_fn` `< Key, Hash_Fn, _↵_Alloc, Comb_Hash_Fn, true >`, [1271](#)
- `__gnu_pbds::detail::ranged_hash_fn` `< Key, null_type, _↵_Alloc, Comb_Hash_Fn, false >`, [1272](#)
- `__gnu_pbds::detail::ranged_hash_fn` `< Key, null_type, _↵_Alloc, Comb_Hash_Fn, true >`, [1273](#)
- `__gnu_pbds::detail::ranged_probe_fn` `< Key, Hash_Fn, _↵_Alloc, Comb_Probe_Fn, Probe_Fn, Store_↵_Hash >`, [1274](#)
- `__gnu_pbds::detail::ranged_probe_fn` `< Key, Hash_Fn, _↵_Alloc, Comb_Probe_Fn, Probe_Fn, false >`, [1274](#)
- `__gnu_pbds::detail::ranged_probe_fn` `< Key, Hash_Fn, _↵_Alloc, Comb_Probe_Fn, Probe_Fn, true >`, [1275](#)
- `__gnu_pbds::detail::ranged_probe_fn` `< Key, null_type, _↵_Alloc, Comb_Probe_Fn, null_type, false >`, [1276](#)
- `__gnu_pbds::detail::rb_tree_map`
 - `node_begin`, [1280](#)
 - `node_end`, [1280](#)
- `__gnu_pbds::detail::rb_tree_map` `< Key, Mapped, Cmp_↵_Fn, Node_And_It_Traits, _Alloc >`, [1277](#)
- `__gnu_pbds::detail::rb_tree_node` `< Value_Type, Meta-↵_data, _Alloc >`, [1280](#)
- `__gnu_pbds::detail::rc` `< _Node, _Alloc >`, [1281](#)
- `__gnu_pbds::detail::rc_binomial_heap` `< Value_Type, Cmp_Fn, _Alloc >`, [1282](#)
- `__gnu_pbds::detail::resize_policy` `< _Tp >`, [1284](#)
- `__gnu_pbds::detail::splay_tree_map`
 - `node_begin`, [1288](#)
 - `node_end`, [1288](#)
- `__gnu_pbds::detail::splay_tree_map` `< Key, Mapped, Cmp_Fn, Node_And_It_Traits, _Alloc >`, [1285](#)
- `__gnu_pbds::detail::splay_tree_node` `< Value_Type, Metadata, _Alloc >`, [1289](#)
- `__gnu_pbds::detail::stored_data` `< _Tv, _Th >`, [1290](#)
- `__gnu_pbds::detail::stored_data` `< _Tv, null_type >`, [1291](#)
- `__gnu_pbds::detail::stored_hash` `< _Th >`, [1292](#)
- `__gnu_pbds::detail::stored_value` `< _Tv >`, [1293](#)
- `__gnu_pbds::detail::synth_access_traits` `< Type_Traits, Set, _ATraits >`, [1293](#)
- `__gnu_pbds::detail::thin_heap` `< Value_Type, Cmp_Fn, _↵_Alloc >`, [1294](#)
- `__gnu_pbds::detail::tree_metadata_helper` `< Node_↵_Update, _BTp >`, [1297](#)
- `__gnu_pbds::detail::tree_metadata_helper` `< Node_↵_Update, false >`, [1297](#)
- `__gnu_pbds::detail::tree_metadata_helper` `< Node_↵_Update, true >`, [1297](#)
- `__gnu_pbds::detail::tree_node_metadata_dispatch` `< Key, Data, Cmp_Fn, Node_Update, _Alloc >`, [1298](#)
- `__gnu_pbds::detail::tree_traits` `< Key, Mapped, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >`, [1299](#)
- `node_const_iterator`, [1299](#)
- `__gnu_pbds::detail::tree_traits` `< Key, null_type, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >`, [1299](#)
- `node_const_iterator`, [1300](#)
- `__gnu_pbds::detail::tree_traits` `< Key, Data, Cmp_Fn, Node_Update, Tag, _Alloc >`, [1298](#)
- `__gnu_pbds::detail::tree_traits` `< Key, Mapped, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >`, [1300](#)
- `node_const_iterator`, [1301](#)
- `__gnu_pbds::detail::tree_traits` `< Key, Mapped, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >`, [1302](#)
- `node_const_iterator`, [1303](#)
- `__gnu_pbds::detail::tree_traits` `< Key, null_type, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >`, [1303](#)
- `node_const_iterator`, [1304](#)
- `__gnu_pbds::detail::tree_traits` `< Key, null_type, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >`, [1305](#)
- `node_const_iterator`, [1306](#)

- `__gnu_pbds::detail::trie_metadata_helper< Node_↵ Update, _BTp >, 1306`
- `__gnu_pbds::detail::trie_metadata_helper< Node_↵ Update, false >, 1306`
- `__gnu_pbds::detail::trie_metadata_helper< Node_↵ Update, true >, 1307`
- `__gnu_pbds::detail::trie_node_metadata_dispatch< Key, Data, Cmp_Fn, Node_Update, _Alloc >, 1307`
- `__gnu_pbds::detail::trie_policy_base< Node_Cltr, Node_ltr, _ATraits, _Alloc >, 1308`
- `__gnu_pbds::detail::trie_traits< Key, Data, _ATraits, Node_Update, Tag, _Alloc >, 1309`
- `__gnu_pbds::detail::trie_traits< Key, Mapped, _ATraits, Node_Update, pat_trie_tag, _Alloc >, 1310`
 - `node_const_iterator, 1310`
 - `node_update, 1310`
 - `synth_access_traits, 1311`
- `__gnu_pbds::detail::trie_traits< Key, null_type, _ATraits, Node_Update, pat_trie_tag, _Alloc >, 1311`
 - `node_const_iterator, 1312`
 - `node_update, 1312`
 - `synth_access_traits, 1312`
- `__gnu_pbds::detail::type_base< Key, Mapped, _Alloc, Store_Hash >, 1313`
- `__gnu_pbds::detail::type_base< Key, Mapped, _Alloc, false >, 1313`
- `__gnu_pbds::detail::type_base< Key, Mapped, _Alloc, true >, 1314`
- `__gnu_pbds::detail::type_base< Key, null_type, _Alloc, false >, 1315`
- `__gnu_pbds::detail::type_base< Key, null_type, _Alloc, true >, 1315`
- `__gnu_pbds::detail::type_dispatch< Key, Mapped, _Alloc, Store_Hash >, 1316`
- `__gnu_pbds::detail::types_traits< Key, Mapped, _Alloc, Store_Hash >, 1316`
- `__gnu_pbds::direct_mask_range_hashing operator(), 1318`
- `__gnu_pbds::direct_mask_range_hashing< Size_Type >, 1317`
- `__gnu_pbds::direct_mod_range_hashing operator(), 1320`
- `__gnu_pbds::direct_mod_range_hashing< Size_Type >, 1319`
- `__gnu_pbds::gp_hash_table`
 - `gp_hash_table, 1322–1324`
- `__gnu_pbds::gp_hash_table< Key, Mapped, Hash_Fn, Eq_Fn, Comb_Probe_Fn, Probe_Fn, Resize_↵ Policy, Store_Hash, _Alloc >, 1320`
- `__gnu_pbds::gp_hash_tag, 1325`
- `__gnu_pbds::hash_exponential_size_policy`
 - `hash_exponential_size_policy, 1326`
- `__gnu_pbds::hash_exponential_size_policy< Size_Type >, 1326`
- `__gnu_pbds::hash_load_check_resize_trigger`
 - `external_load_access, 1328`
 - `get_loads, 1328`
 - `hash_load_check_resize_trigger, 1328`
 - `notify_cleared, 1328`
 - `notify_inserted, 1328`
 - `notify_resized, 1329`
 - `set_loads, 1329`
- `__gnu_pbds::hash_load_check_resize_trigger< External_↵ _Load_Access, Size_Type >, 1327`
- `__gnu_pbds::hash_prime_size_policy, 1329`
 - `hash_prime_size_policy, 1330`
 - `size_type, 1330`
- `__gnu_pbds::hash_standard_resize_policy`
 - `get_actual_size, 1332`
 - `get_new_size, 1332`
 - `get_size_policy, 1332`
 - `get_trigger_policy, 1333`
 - `hash_standard_resize_policy, 1331, 1332`
 - `resize, 1333`
- `__gnu_pbds::hash_standard_resize_policy< Size_Policy, Trigger_Policy, External_Size_Access, Size_↵ Type >, 1330`
- `__gnu_pbds::insert_error, 1334`
 - `what, 1334`
- `__gnu_pbds::join_error, 1335`
 - `what, 1335`
- `__gnu_pbds::linear_probe_fn`
 - `operator(), 1336`
- `__gnu_pbds::linear_probe_fn< Size_Type >, 1336`
- `__gnu_pbds::list_update`
 - `list_update, 1337`
- `__gnu_pbds::list_update< Key, Mapped, Eq_Fn, Update_Policy, _Alloc >, 1336`
- `__gnu_pbds::list_update_tag, 1338`
- `__gnu_pbds::lu_counter_policy`
 - `max_count, 1340`
 - `metadata_reference, 1339`
 - `metadata_type, 1339`
 - `operator(), 1340`
- `__gnu_pbds::lu_counter_policy< Max_Count, _Alloc >, 1338`
- `__gnu_pbds::lu_move_to_front_policy`
 - `metadata_reference, 1341`
 - `metadata_type, 1341`
 - `operator(), 1341`
- `__gnu_pbds::lu_move_to_front_policy< _Alloc >, 1340`
- `__gnu_pbds::null_node_update< _Tp1, _Tp2, _Tp3, _↵ Tp4 >, 1342`
- `__gnu_pbds::null_type, 1342`
- `__gnu_pbds::ov_tree_tag, 1343`
- `__gnu_pbds::pairing_heap_tag, 1344`
- `__gnu_pbds::pat_trie_tag, 1345`
- `__gnu_pbds::point_invalidation_guarantee, 1346`

- `__gnu_pbds::priority_queue< _Tv, Cmp_Fn, Tag, _Alloc`
 `>, 1346`
- `__gnu_pbds::priority_queue_tag, 1348`
- `__gnu_pbds::quadratic_probe_fn`
 `operator(), 1349`
- `__gnu_pbds::quadratic_probe_fn< Size_Type >, 1348`
- `__gnu_pbds::range_invalidation_guarantee, 1349`
- `__gnu_pbds::rb_tree_tag, 1350`
- `__gnu_pbds::rc_binomial_heap_tag, 1351`
- `__gnu_pbds::resize_error, 1352`
 `what, 1352`
- `__gnu_pbds::sample_probe_fn, 1353`
 `operator(), 1353`
 `sample_probe_fn, 1353`
 `swap, 1353`
- `__gnu_pbds::sample_range_hashing, 1354`
 `notify_resized, 1355`
 `operator(), 1355`
 `sample_range_hashing, 1354`
 `size_type, 1354`
 `swap, 1355`
- `__gnu_pbds::sample_ranged_hash_fn, 1355`
 `notify_resized, 1356`
 `operator(), 1356`
 `sample_ranged_hash_fn, 1356`
 `swap, 1356`
- `__gnu_pbds::sample_ranged_probe_fn, 1356`
- `__gnu_pbds::sample_resize_policy, 1357`
 `get_new_size, 1358`
 `is_resize_needed, 1358`
 `notify_cleared, 1358`
 `notify_erase_search_collision, 1358`
 `notify_erase_search_end, 1358`
 `notify_erase_search_start, 1358`
 `notify_erased, 1358`
 `notify_find_search_collision, 1358`
 `notify_find_search_end, 1359`
 `notify_find_search_start, 1359`
 `notify_insert_search_collision, 1359`
 `notify_insert_search_end, 1359`
 `notify_insert_search_start, 1359`
 `notify_inserted, 1359`
 `notify_resized, 1359`
 `sample_range_hashing, 1359`
 `sample_resize_policy, 1358`
 `size_type, 1358`
 `swap, 1359`
- `__gnu_pbds::sample_resize_trigger, 1360`
 `is_grow_needed, 1361`
 `is_resize_needed, 1361`
 `notify_cleared, 1361`
 `notify_erase_search_collision, 1361`
 `notify_erase_search_end, 1361`
 `notify_erase_search_start, 1361`
 `notify_erased, 1361`
 `notify_externally_resized, 1361`
 `notify_find_search_collision, 1361`
 `notify_find_search_end, 1361`
 `notify_find_search_start, 1362`
 `notify_insert_search_collision, 1362`
 `notify_insert_search_end, 1362`
 `notify_insert_search_start, 1362`
 `notify_inserted, 1362`
 `notify_resized, 1362`
 `sample_range_hashing, 1362`
 `sample_resize_trigger, 1361`
 `size_type, 1360`
 `swap, 1362`
- `__gnu_pbds::sample_size_policy, 1362`
 `get_nearest_larger_size, 1363`
 `get_nearest_smaller_size, 1363`
 `sample_range_hashing, 1363`
 `sample_size_policy, 1363`
 `size_type, 1363`
 `swap, 1364`
- `__gnu_pbds::sample_tree_node_update< Const_Node↵`
 `_Iter, Node_Iter, Cmp_Fn, _Alloc >, 1364`
- `__gnu_pbds::sample_trie_access_traits, 1364`
 `begin, 1365`
 `e_pos, 1365`
 `e_type, 1365`
 `end, 1365`
- `__gnu_pbds::sample_trie_node_update`
 `operator(), 1366`
 `sample_trie_node_update, 1366`
- `__gnu_pbds::sample_trie_node_update< Node_Cltr,`
 `Node_Itr, ATraits, _Alloc >, 1365`
- `__gnu_pbds::sample_update_policy, 1366`
 `metadata_type, 1367`
 `operator(), 1367`
 `sample_update_policy, 1367`
 `swap, 1367`
- `__gnu_pbds::sequence_tag, 1368`
- `__gnu_pbds::splay_tree_tag, 1369`
- `__gnu_pbds::string_tag, 1370`
- `__gnu_pbds::thin_heap_tag, 1371`
- `__gnu_pbds::tree`
 `cmp_fn, 1372`
 `tree, 1373`
- `__gnu_pbds::tree< Key, Mapped, Cmp_Fn, Tag, Node↵`
 `Update, _Alloc >, 1371`
- `__gnu_pbds::tree_order_statistics_node_update`
 `find_by_order, 1375`
 `operator(), 1375`
 `order_of_key, 1376`
- `__gnu_pbds::tree_order_statistics_node_update< Node↵`
 `_Cltr, Node_Itr, Cmp_Fn, _Alloc >, 1374`
- `__gnu_pbds::tree_tag, 1376`

- __gnu_pbds::trie
 - access_traits, [1378](#)
 - trie, [1378](#)
- __gnu_pbds::trie< Key, Mapped, _ATraits, Tag, Node_↔
 - Update, _Alloc >, [1377](#)
- __gnu_pbds::trie_order_statistics_node_update
 - find_by_order, [1381](#)
 - operator(), [1381](#)
 - order_of_key, [1381](#)
 - order_of_prefix, [1381](#)
- __gnu_pbds::trie_order_statistics_node_update< Node_↔
 - _Cltr, Node_ltr, _ATraits, _Alloc >, [1379](#)
- __gnu_pbds::trie_prefix_search_node_update
 - a_const_iterator, [1384](#)
 - access_traits, [1384](#)
 - allocator_type, [1384](#)
 - operator(), [1384](#)
 - prefix_range, [1384](#), [1385](#)
 - size_type, [1384](#)
- __gnu_pbds::trie_prefix_search_node_update< Node_↔
 - _Cltr, Node_ltr, _ATraits, _Alloc >, [1382](#)
- __gnu_pbds::trie_string_access_traits
 - begin, [1387](#)
 - const_iterator, [1386](#)
 - e_pos, [1387](#)
 - e_type, [1387](#)
 - end, [1387](#)
- __gnu_pbds::trie_string_access_traits< String, Min_E_↔
 - Val, Max_E_Val, Reverse, _Alloc >, [1386](#)
- __gnu_pbds::trie_tag, [1388](#)
- __gnu_pbds::trivial_iterator_tag, [1389](#)
- __gnu_profile, [453](#)
 - _GLIBCXX_PROFILE_DEFINE_UNINIT_DATA, [457](#)
 - _env_t, [457](#)
 - _profcxx_init, [457](#)
 - _report, [457](#)
- __gnu_profile::__container_size_info, [1389](#)
- __gnu_profile::__container_size_stack_info, [1391](#)
- __gnu_profile::__hashfunc_info, [1392](#)
- __gnu_profile::__hashfunc_stack_info, [1393](#)
- __gnu_profile::__list2vector_info, [1394](#)
- __gnu_profile::__map2umap_info, [1395](#)
- __gnu_profile::__map2umap_stack_info, [1396](#)
- __gnu_profile::__object_info_base, [1397](#)
- __gnu_profile::__reentrance_guard, [1398](#)
- __gnu_profile::__stack_hash, [1399](#)
- __gnu_profile::__trace_base< __object_info, __stack_↔
 - info >, [1399](#)
- __gnu_profile::__trace_container_size, [1400](#)
- __gnu_profile::__trace_hash_func, [1401](#)
- __gnu_profile::__trace_hashtable_size, [1402](#)
- __gnu_profile::__trace_map2umap, [1403](#)
- __gnu_profile::__trace_vector_size, [1404](#)
- __gnu_profile::__trace_vector_to_list, [1405](#)
- __gnu_profile::__vector2list_info, [1406](#)
- __gnu_profile::__vector2list_stack_info, [1407](#)
- __gnu_profile::__warning_data, [1408](#)
- __gnu_sequential, [457](#)
- __heap_select
 - std, [563](#)
- __init_winner
 - __gnu_parallel::_LoserTree< false, _Tp, _Compare
 - >, [1070](#)
- __inner_product_selector
 - __gnu_parallel::__inner_product_selector, [1032](#)
- __inplace_stable_sort
 - std, [563](#)
- __insert_start
 - __gnu_parallel::_LoserTree, [1067](#)
 - __gnu_parallel::_LoserTree< false, _Tp, _Compare
 - >, [1070](#)
 - __gnu_parallel::_LoserTreeBase, [1073](#)
- __insertion_sort
 - std, [564](#)
- __introsort_loop
 - std, [564](#)
- __invoke
 - std, [564](#)
- __iointit
 - std, [645](#)
- __is_sorted
 - __gnu_parallel, [419](#)
- __iterator_category
 - Iterators, [108](#)
- __lg
 - std, [564](#)
- __match_flag
 - std::regex_constants, [694](#)
- __median
 - SGL, [302](#), [303](#)
- __median_of_three_iterators
 - __gnu_parallel, [419](#)
- __merge_adaptive
 - std, [564](#)
- __merge_advance
 - __gnu_parallel, [419](#)
- __merge_advance_movc
 - __gnu_parallel, [420](#)
- __merge_advance_usual
 - __gnu_parallel, [420](#)
- __merge_without_buffer
 - std, [565](#)
- __move_median_to_first
 - std, [565](#)
- __move_merge
 - std, [565](#)
- __move_merge_adaptive
 - std, [565](#)

- `__move_merge_adaptive_backward`
 - `std`, 565
- `__new_val`
 - `__gnu_parallel::__replace_if_selector`, 1041
 - `__gnu_parallel::__replace_selector`, 1043
- `__num_bitmaps`
 - `__gnu_cxx::__detail`, 389
- `__num_blocks`
 - `__gnu_cxx::__detail`, 389
- `__num_get_type`
 - `std::basic_ios`, 1836
 - `std::basic_ofstream`, 2022
 - `std::basic_ostream`, 2062
 - `std::basic_ostringstream`, 2101
- `__num_put_type`
 - `std::basic_fstream`, 1732
 - `std::basic_ifstream`, 1790
 - `std::basic_ios`, 1836
 - `std::basic_iostream`, 1867
 - `std::basic_istream`, 1926
 - `std::basic_istreamstream`, 1974
 - `std::basic_stringstream`, 2247
- `__once_call`
 - `std`, 645
- `__once_callable`
 - `std`, 646
- `__once_proxy`
 - `std`, 566
- `__parallel_merge_advance`
 - `__gnu_parallel`, 421
- `__parallel_nth_element`
 - `__gnu_parallel`, 422
- `__parallel_partial_sort`
 - `__gnu_parallel`, 422
- `__parallel_partial_sum`
 - `__gnu_parallel`, 424
- `__parallel_partial_sum_basecase`
 - `__gnu_parallel`, 424
- `__parallel_partial_sum_linear`
 - `__gnu_parallel`, 425
- `__parallel_partition`
 - `__gnu_parallel`, 425
- `__parallel_random_shuffle`
 - `__gnu_parallel`, 426
- `__parallel_random_shuffle_drs`
 - `__gnu_parallel`, 426
- `__parallel_random_shuffle_drs_pu`
 - `__gnu_parallel`, 427
- `__parallel_sort`
 - `__gnu_parallel`, 427–431
- `__parallel_sort_qs`
 - `__gnu_parallel`, 432
- `__parallel_sort_qs_conquer`
 - `__gnu_parallel`, 432
- `__parallel_sort_qs_divide`
 - `__gnu_parallel`, 433
- `__parallel_sort_qsb`
 - `__gnu_parallel`, 433
- `__parallel_unique_copy`
 - `__gnu_parallel`, 433, 435
- `__partition`
 - `std`, 566
- `__polynomial`
 - `std::regex_constants`, 700
- `__profcxx_init`
 - `__gnu_profile`, 457
- `__ptr_rebind`
 - `std`, 559
- `__qsb_conquer`
 - `__gnu_parallel`, 435
- `__qsb_divide`
 - `__gnu_parallel`, 436
- `__qsb_local_sort_with_helping`
 - `__gnu_parallel`, 436
- `__random_number_pow2`
 - `__gnu_parallel`, 436
- `__rd_log2`
 - `__gnu_parallel`, 437
- `__rebind_m`
 - `__gnu_pbds::detail::pat_trie_base::_Node_citer`, 1262
 - `__gnu_pbds::detail::pat_trie_base::_Node_iter`, 1265
- `__replace_if_selector`
 - `__gnu_parallel::__replace_if_selector`, 1040
- `__replace_selector`
 - `__gnu_parallel::__replace_selector`, 1042
- `__report`
 - `__gnu_profile`, 457
- `__reverse`
 - `std`, 567
- `__rotate`
 - `std::algo.h`, 3762
- `__rotate_adaptive`
 - `std`, 567
- `__round_up_to_pow2`
 - `__gnu_parallel`, 437
- `__sample`
 - `experimental/algorithm`, 3403
- `__search_n_aux`
 - `std`, 567
- `__search_template`
 - `__gnu_parallel`, 437
- `__sequential_multiway_merge`
 - `__gnu_parallel`, 438
- `__sequential_random_shuffle`
 - `__gnu_parallel`, 438
- `__shared_timed_mutex_base`
 - `Mutexes`, 169

- __shrink
 - __gnu_parallel, 439
- __shrink_and_double
 - __gnu_parallel, 439
- __stable_partition_adaptive
 - std, 568
- __static_pointer_cast
 - __gnu_cxx, 375
- __streambuf_type
 - std::basic_streambuf, 2148
 - std::wbuffer_convert, 3364
- __syntax_option
 - std::regex_constants, 694
- __try_to_lock
 - std, 568
- __umap_traits
 - std, 559
- __ummap_traits
 - std, 559
- __umset_traits
 - std, 559
- __unguarded_insertion_sort
 - std, 568
- __unguarded_linear_insert
 - std, 569
- __unguarded_partition
 - std, 569
- __unguarded_partition_pivot
 - std, 569
- __unique_copy
 - std, 569, 570
- __uset_traits
 - std, 559
- __valid_range
 - __gnu_debug, 399, 400
- __valid_range_aux
 - __gnu_debug, 400
- __verbose_terminate_handler
 - Exceptions, 74
- __versa_string
 - __gnu_cxx::__versa_string, 738–741
- __yield
 - __gnu_parallel, 440
- ~_LoserTreeBase
 - __gnu_parallel::_LoserTreeBase, 1073
- ~_RestrictedBoundedConcurrentQueue
 - __gnu_parallel::_RestrictedBoundedConcurrentQueue, 1100
- ~_Safe_sequence_base
 - __gnu_debug::_Safe_sequence_base, 972
- ~_Safe_unordered_container_base
 - __gnu_debug::_Safe_unordered_container_base, 980
- ~__allocated_ptr
 - std::__allocated_ptr, 1424
- ~__versa_string
 - __gnu_cxx::__versa_string, 741
- ~any
 - std::experimental::fundamentals_v1::any, 2529
- ~auto_ptr
 - std::auto_ptr, 1686
- ~basic_filebuf
 - std::basic_filebuf, 1702
- ~basic_fstream
 - std::basic_fstream, 1735
- ~basic_ifstream
 - std::basic_ifstream, 1793
- ~basic_ios
 - std::basic_ios, 1839
- ~basic_iostream
 - std::basic_iostream, 1869
- ~basic_istream
 - std::basic_istream, 1928
- ~basic_istreamstream
 - std::basic_istreamstream, 1977
- ~basic_ofstream
 - std::basic_ofstream, 2025
- ~basic_ostream
 - std::basic_ostream, 2065
- ~basic_ostringstream
 - std::basic_ostringstream, 2104
- ~basic_regex
 - std::basic_regex, 2139
- ~basic_streambuf
 - std::basic_streambuf, 2149
- ~basic_string
 - std::basic_string, 2173
- ~basic_stringstream
 - std::basic_stringstream, 2251
- ~collate
 - std::collate, 2378
- ~ctype
 - std::ctype< char >, 2415
 - std::ctype< wchar_t >, 2429
- ~deque
 - std::deque, 2486
- ~facet
 - std::locale::facet, 2767
- ~forward_list
 - std::forward_list, 2561
- ~gslice
 - Numeric Arrays, 203
- ~ios_base
 - std::ios_base, 2659
- ~locale
 - std::locale, 2760
- ~match_results
 - std::match_results, 2814

- ~messages
 - std::messages, [2832](#)
- ~money_get
 - std::money_get, [2844](#)
- ~money_put
 - std::money_put, [2848](#)
- ~moneypunct
 - std::moneypunct, [2855](#)
- ~num_get
 - std::num_get, [2933](#)
- ~num_put
 - std::num_put, [2950](#)
- ~numpunct
 - std::numpunct, [2989](#)
- ~sentry
 - std::basic_ostream::sentry, [2095](#)
- ~stdio_filebuf
 - __gnu_cxx::stdio_filebuf, [874](#)
- ~temporary_buffer
 - __gnu_cxx::temporary_buffer, [917](#)
- ~time_get
 - std::time_get, [3153](#)
- ~time_put
 - std::time_put, [3176](#)
- ~type_info
 - std::type_info, [3220](#)
- ~unique_ptr
 - std::unique_ptr, [3235](#)
 - std::unique_ptr< _Tp[], _Dp >, [3241](#)
- ~vector
 - std::vector, [3344](#)
- a
 - std::extreme_value_distribution, [2548](#)
 - std::weibull_distribution, [3381](#)
- a_const_iterator
 - __gnu_pbds::trie_prefix_search_node_update, [1384](#)
- ATOMIC_BOOL_LOCK_FREE
 - Atomics, [18](#)
- abi, [458](#)
- abs
 - Complex Numbers, [49](#)
- access_traits
 - __gnu_pbds::trie, [1378](#)
 - __gnu_pbds::trie_prefix_search_node_update, [1384](#)
- accumulate
 - std, [571](#)
- accumulate_minimal_n
 - __gnu_parallel::_Settings, [1103](#)
- acos
 - std, [572](#)
- acosh
 - std, [572](#)
- Adaptors for pointers to functions, [6](#)
- ptr_fun, [6](#)
- Adaptors for pointers to members, [7](#)
- addressof
 - Utilities, [353](#)
- adjacent_difference
 - std, [572](#)
- adjacent_difference_minimal_n
 - __gnu_parallel::_Settings, [1103](#)
- adjacent_find
 - Non-Mutating, [173](#), [174](#)
- adjustfield
 - std::basic_fstream, [1777](#)
 - std::basic_ifstream, [1826](#)
 - std::basic_ios, [1854](#)
 - std::basic_iostream, [1913](#)
 - std::basic_istream, [1960](#)
 - std::basic_istreamstream, [2010](#)
 - std::basic_ofstream, [2051](#)
 - std::basic_ostream, [2088](#)
 - std::basic_ostreamstream, [2128](#)
 - std::basic_stringstream, [2293](#)
 - std::ios_base, [2665](#)
- adopt_lock
 - Mutexes, [169](#)
- advance
 - std, [573](#)
- algo.h, [3388](#)
- algbase.h, [3398](#)
- algorithm, [3400](#), [3402](#)
- algorithmfwd.h, [3403](#), [3409](#)
- Algorithms, [8](#)
- align
 - std, [573](#)
- aligned_buffer.h, [3417](#)
- all
 - std::bitset, [2319](#)
 - std::locale, [2764](#)
 - std::tr2::dynamic_bitset, [3195](#)
- all_of
 - Non-Mutating, [174](#)
- alloc_traits.h, [3417](#), [3418](#)
- allocate
 - __gnu_cxx::__alloc_traits, [716](#)
 - std::allocator_traits, [1651](#), [1652](#)
 - std::allocator_traits< allocator< _Tp > >, [1656](#), [1657](#)
- allocate_shared
 - Pointer Abstractions, [235](#)
 - std::shared_ptr, [3122](#)
- allocated_ptr.h, [3419](#)
- allocator.h, [3419](#)
- allocator_type
 - __gnu_pbds::trie_prefix_search_node_update, [1384](#)
 - std::allocator_traits, [1649](#)

- std::allocator_traits< allocator< _Tp > >, 1655
- std::set, 3084
- std::unordered_map, 3246
- std::unordered_multimap, 3270
- std::unordered_multiset, 3292
- std::unordered_set, 3314
- Allocators, 9
 - __allocator_base, 10
- alpha
 - std::gamma_distribution, 2598
- any, 3420
 - std::bitset, 2319
 - std::experimental::fundamentals_v1::any, 2528, 2529
 - std::tr2::dynamic_bitset, 3195
- any_cast
 - Type-safe container of any type, 343–345
- any_of
 - Non-Mutating, 175
- app
 - std::basic_fstream, 1778
 - std::basic_ifstream, 1826
 - std::basic_ios, 1854
 - std::basic_iostream, 1914
 - std::basic_istream, 1960
 - std::basic_istreamstream, 2010
 - std::basic_ofstream, 2051
 - std::basic_ostream, 2088
 - std::basic_ostreamstream, 2128
 - std::basic_stringstream, 2293
 - std::ios_base, 2665
- append
 - __gnu_cxx::__versa_string, 741, 742, 744, 745
 - __gnu_debug::basic_string, 990, 991
 - std::basic_string, 2173–2176
 - std::tr2::dynamic_bitset, 3195
- apply
 - Numeric Arrays, 203, 204
- apply_generator
 - __gnu_cxx::typelist, 390
- arg
 - Complex Numbers, 49
 - std, 574
- argument_type
 - __gnu_cxx::__detail::_Ffit_finder, 721
 - __gnu_cxx::binary_compose, 805
 - __gnu_cxx::select1st, 866
 - __gnu_cxx::select2nd, 867
 - __gnu_cxx::subtractive_rng, 915
 - __gnu_cxx::unary_compose, 928
 - __gnu_parallel::__binder1st, 1015
 - __gnu_parallel::__binder2nd, 1016
 - __gnu_parallel::__unary_negate, 1048
 - std::Maybe_unary_or_binary_function< _Res, _T1 >, 1625
- std::binder1st, 2308
- std::binder2nd, 2309
- std::const_mem_fun_ref_t, 2396
- std::const_mem_fun_t, 2397
- std::hash< __gnu_cxx::throw_value_limit >, 2616
- std::hash< __gnu_cxx::throw_value_random >, 2617
- std::logical_not, 2774
- std::mem_fun_ref_t, 2823
- std::mem_fun_t, 2824
- std::negate, 2918
- std::pointer_to_unary_function, 3028
- std::unary_function, 3222
- std::unary_negate, 3223
- Arithmetic Classes, 11
- array, 3421–3423
- Array creation functions, 12
- array_allocator.h, 3423
- asin
 - std, 574
- asinh
 - std, 574
- assertions.h, 3424
- assign
 - __gnu_cxx::__versa_string, 745–748
 - __gnu_debug::basic_string, 991, 993
 - std::basic_regex, 2139–2141
 - std::basic_string, 2176–2179
 - std::deque, 2490, 2491
 - std::forward_list, 2562
 - std::list, 2738, 2739
 - std::vector, 3345
- assoc_container.hpp, 3424
- assoc_laguerre
 - Mathematical Special Functions, 117, 141
- assoc_laguerref
 - Mathematical Special Functions, 118
- assoc_laguerrel
 - Mathematical Special Functions, 118
- assoc_legendre
 - Mathematical Special Functions, 118, 141
- assoc_legendref
 - Mathematical Special Functions, 119
- assoc_legendrel
 - Mathematical Special Functions, 119
- Associative, 13
- async
 - Futures, 87
- at
 - __gnu_cxx::__versa_string, 749
 - __gnu_debug::basic_string, 994
 - std::basic_string, 2180
 - std::deque, 2491, 2492
 - std::map, 2788

- [std::unordered_map](#), [3251](#)
 - [std::vector](#), [3346](#)
- [atan](#)
 - [std](#), [574](#)
- [atanh](#)
 - [std](#), [574](#)
- [ate](#)
 - [std::basic_fstream](#), [1778](#)
 - [std::basic_ifstream](#), [1826](#)
 - [std::basic_ios](#), [1854](#)
 - [std::basic_iostream](#), [1914](#)
 - [std::basic_istream](#), [1960](#)
 - [std::basic_istreamstream](#), [2011](#)
 - [std::basic_ofstream](#), [2051](#)
 - [std::basic_ostream](#), [2088](#)
 - [std::basic_ostreamstream](#), [2129](#)
 - [std::basic_stringstream](#), [2293](#)
 - [std::ios_base](#), [2665](#)
- [atomic](#), [3425](#)
- [atomic_base.h](#), [3429](#)
- [atomic_bool](#)
 - [Atomics](#), [18](#)
- [atomic_char](#)
 - [Atomics](#), [18](#)
- [atomic_char16_t](#)
 - [Atomics](#), [19](#)
- [atomic_char32_t](#)
 - [Atomics](#), [19](#)
- [atomic_compare_exchange_strong](#)
 - [Pointer Abstractions](#), [235](#)
- [atomic_compare_exchange_strong_explicit](#)
 - [Pointer Abstractions](#), [236](#)
- [atomic_compare_exchange_weak](#)
 - [Pointer Abstractions](#), [237](#)
- [atomic_compare_exchange_weak_explicit](#)
 - [Pointer Abstractions](#), [238](#)
- [atomic_exchange](#)
 - [Pointer Abstractions](#), [239](#)
- [atomic_exchange_explicit](#)
 - [Pointer Abstractions](#), [240](#)
- [atomic_futex.h](#), [3430](#)
- [atomic_int](#)
 - [Atomics](#), [19](#)
- [atomic_int_fast16_t](#)
 - [Atomics](#), [19](#)
- [atomic_int_fast32_t](#)
 - [Atomics](#), [19](#)
- [atomic_int_fast64_t](#)
 - [Atomics](#), [19](#)
- [atomic_int_fast8_t](#)
 - [Atomics](#), [19](#)
- [atomic_int_least16_t](#)
 - [Atomics](#), [19](#)
- [atomic_int_least32_t](#)
 - [Atomics](#), [20](#)
- [atomic_int_least64_t](#)
 - [Atomics](#), [20](#)
- [atomic_int_least8_t](#)
 - [Atomics](#), [20](#)
- [atomic_intmax_t](#)
 - [Atomics](#), [20](#)
- [atomic_intptr_t](#)
 - [Atomics](#), [20](#)
- [atomic_is_lock_free](#)
 - [Pointer Abstractions](#), [241](#)
- [atomic_llong](#)
 - [Atomics](#), [20](#)
- [atomic_load](#)
 - [Pointer Abstractions](#), [241](#), [242](#)
- [atomic_load_explicit](#)
 - [Pointer Abstractions](#), [242](#)
- [atomic_lockfree_defines.h](#), [3430](#)
- [atomic_long](#)
 - [Atomics](#), [20](#)
- [atomic_ptrdiff_t](#)
 - [Atomics](#), [20](#)
- [atomic_schar](#)
 - [Atomics](#), [21](#)
- [atomic_short](#)
 - [Atomics](#), [21](#)
- [atomic_size_t](#)
 - [Atomics](#), [21](#)
- [atomic_store](#)
 - [Pointer Abstractions](#), [243](#)
- [atomic_store_explicit](#)
 - [Pointer Abstractions](#), [244](#)
- [atomic_uchar](#)
 - [Atomics](#), [21](#)
- [atomic_uint](#)
 - [Atomics](#), [21](#)
- [atomic_uint_fast16_t](#)
 - [Atomics](#), [21](#)
- [atomic_uint_fast32_t](#)
 - [Atomics](#), [21](#)
- [atomic_uint_fast64_t](#)
 - [Atomics](#), [21](#)
- [atomic_uint_fast8_t](#)
 - [Atomics](#), [22](#)
- [atomic_uint_least16_t](#)
 - [Atomics](#), [22](#)
- [atomic_uint_least32_t](#)
 - [Atomics](#), [22](#)
- [atomic_uint_least64_t](#)
 - [Atomics](#), [22](#)
- [atomic_uint_least8_t](#)
 - [Atomics](#), [22](#)
- [atomic_uintmax_t](#)
 - [Atomics](#), [22](#)

- atomic_uintptr_t
 - Atomics, [22](#)
- atomic_ullong
 - Atomics, [22](#)
- atomic_ulong
 - Atomics, [23](#)
- atomic_ushort
 - Atomics, [23](#)
- atomic_wchar_t
 - Atomics, [23](#)
- atomic_word.h, [3431](#)
- atomicity.h, [3431](#)
- Atomics, [14](#)
 - ATOMIC_BOOL_LOCK_FREE, [18](#)
 - atomic_bool, [18](#)
 - atomic_char, [18](#)
 - atomic_char16_t, [19](#)
 - atomic_char32_t, [19](#)
 - atomic_int, [19](#)
 - atomic_int_fast16_t, [19](#)
 - atomic_int_fast32_t, [19](#)
 - atomic_int_fast64_t, [19](#)
 - atomic_int_fast8_t, [19](#)
 - atomic_int_least16_t, [19](#)
 - atomic_int_least32_t, [20](#)
 - atomic_int_least64_t, [20](#)
 - atomic_int_least8_t, [20](#)
 - atomic_intmax_t, [20](#)
 - atomic_intptr_t, [20](#)
 - atomic_llong, [20](#)
 - atomic_long, [20](#)
 - atomic_ptrdiff_t, [20](#)
 - atomic_schar, [21](#)
 - atomic_short, [21](#)
 - atomic_size_t, [21](#)
 - atomic_uchar, [21](#)
 - atomic_uint, [21](#)
 - atomic_uint_fast16_t, [21](#)
 - atomic_uint_fast32_t, [21](#)
 - atomic_uint_fast64_t, [21](#)
 - atomic_uint_fast8_t, [22](#)
 - atomic_uint_least16_t, [22](#)
 - atomic_uint_least32_t, [22](#)
 - atomic_uint_least64_t, [22](#)
 - atomic_uint_least8_t, [22](#)
 - atomic_uintmax_t, [22](#)
 - atomic_uintptr_t, [22](#)
 - atomic_ullong, [22](#)
 - atomic_ulong, [23](#)
 - atomic_ushort, [23](#)
 - atomic_wchar_t, [23](#)
 - kill_dependency, [23](#)
 - memory_order, [23](#)
- auto_ptr
 - std::auto_ptr, [1685](#), [1686](#)
- auto_ptr.h, [3432](#)
- awk
 - std::regex_constants, [700](#)
- b
 - std::extreme_value_distribution, [2548](#)
 - std::weibull_distribution, [3381](#)
- back
 - __gnu_cxx::__versa_string, [750](#)
 - __gnu_debug::basic_string, [995](#)
 - std::basic_string, [2181](#)
 - std::deque, [2492](#)
 - std::list, [2739](#), [2740](#)
 - std::queue, [3044](#)
 - std::vector, [3347](#)
- back_insert_iterator
 - std::back_insert_iterator, [1692](#)
- back_inserter
 - Iterators, [108](#)
- backward_warning.h, [3432](#)
- bad
 - std::basic_fstream, [1736](#)
 - std::basic_ifstream, [1793](#)
 - std::basic_ios, [1840](#)
 - std::basic_iostream, [1870](#)
 - std::basic_istream, [1929](#)
 - std::basic_istreamstream, [1978](#)
 - std::basic_ofstream, [2026](#)
 - std::basic_ostream, [2066](#)
 - std::basic_ostreamstream, [2105](#)
 - std::basic_stringstream, [2252](#)
- badbit
 - std::basic_fstream, [1778](#)
 - std::basic_ifstream, [1827](#)
 - std::basic_ios, [1854](#)
 - std::basic_iostream, [1914](#)
 - std::basic_istream, [1961](#)
 - std::basic_istreamstream, [2011](#)
 - std::basic_ofstream, [2051](#)
 - std::basic_ostream, [2089](#)
 - std::basic_ostreamstream, [2129](#)
 - std::basic_stringstream, [2294](#)
 - std::ios_base, [2665](#)
- balanced_quicksort.h, [3432](#)
- base
 - __gnu_debug::_Safe_iterator, [941](#)
 - __gnu_debug::_Safe_local_iterator, [955](#)
 - std::discard_block_engine, [2506](#)
 - std::independent_bits_engine, [2640](#)
 - std::reverse_iterator, [3073](#)
 - std::shuffle_order_engine, [3126](#)
- Base and Implementation Classes, [24](#), [26](#)
 - _Opcode, [25](#)

Base and Policy Classes, 29–31

base.h, 3433

basefield

std::basic_fstream, 1778

std::basic_ifstream, 1827

std::basic_ios, 1854

std::basic_iostream, 1914

std::basic_istream, 1961

std::basic_istreamstream, 2011

std::basic_ofstream, 2052

std::basic_ostream, 2089

std::basic_ostreamstream, 2129

std::basic_stringstream, 2294

std::ios_base, 2665

basic

std::regex_constants, 700

basic_file.h, 3434

basic_filebuf

std::basic_filebuf, 1702

basic_fstream

std::basic_fstream, 1734, 1735

basic_ifstream

std::basic_ifstream, 1792, 1793

basic_ios

std::basic_ios, 1839, 1840

basic_ios.h, 3435

basic_ios.tcc, 3435

basic_iostream

std::basic_iostream, 1869

basic_istream

std::basic_istream, 1928

basic_istreamstream

std::basic_istreamstream, 1977

basic_iterator.h, 3435

basic_ofstream

std::basic_ofstream, 2024, 2025

basic_ostream

std::basic_ostream, 2065

basic_ostreamstream

std::basic_ostreamstream, 2104

basic_regex

std::basic_regex, 2136–2138

basic_streambuf

std::basic_streambuf, 2149

basic_string

std::basic_string, 2170–2173

basic_string.h, 3436

basic_string.tcc, 3438

basic_stringbuf

std::basic_stringbuf, 2223

basic_stringstream

std::basic_stringstream, 2249

before_begin

std::forward_list, 2563

beg

std::basic_fstream, 1778

std::basic_ifstream, 1827

std::basic_ios, 1854

std::basic_iostream, 1914

std::basic_istream, 1961

std::basic_istreamstream, 2011

std::basic_ofstream, 2052

std::basic_ostream, 2089

std::basic_ostreamstream, 2129

std::basic_stringstream, 2294

std::ios_base, 2665

begin

__gnu_cxx::__versa_string, 750

__gnu_cxx::temporary_buffer, 917

__gnu_parallel::PseudoSequence, 1094

__gnu_pbds::sample_trie_access_traits, 1365

__gnu_pbds::trie_string_access_traits, 1387

Numeric Arrays, 204

std, 574, 575

std::_Temporary_buffer, 1634

std::basic_string, 2181

std::deque, 2492, 2493

std::forward_list, 2563

std::list, 2740

std::map, 2789

std::match_results, 2814

std::multimap, 2877

std::multiset, 2902

std::set, 3089

std::unordered_map, 3252

std::unordered_multimap, 3275

std::unordered_multiset, 3296, 3297

std::unordered_set, 3318, 3320

std::vector, 3347

Bernoulli Distributions, 32

operator!=, 33

operator<<, 33, 34

operator>>, 34

bernoulli_distribution

std::bernoulli_distribution, 2300

beta

Mathematical Special Functions, 119, 141

std::gamma_distribution, 2598

betaf

Mathematical Special Functions, 120

betal

Mathematical Special Functions, 120

bin_search_tree.hpp, 3439

binary

std::basic_fstream, 1779

std::basic_ifstream, 1827

std::basic_ios, 1855

std::basic_iostream, 1915

- std::basic_istream, 1961
- std::basic_istream, 2011
- std::basic_ofstream, 2052
- std::basic_ostream, 2089
- std::basic_ostream, 2129
- std::basic_stringstream, 2294
- std::ios_base, 2666
- Binary Search, 36
 - binary_search, 37
 - equal_range, 37, 38
 - lower_bound, 38, 39
 - upper_bound, 39, 40
- binary_heap.hpp, 3440
- binary_heap_const_iterator_
 - __gnu_pbds::detail::binary_heap_const_iterator_, 1174
- binary_heap_point_const_iterator_
 - __gnu_pbds::detail::binary_heap_point_const_iterator_, 1178
- binary_search
 - Binary Search, 37
- bind
 - Binder Classes, 42
- bind1st
 - Binder Classes, 42
- bind2nd
 - Binder Classes, 42
- Binder Classes, 41
 - bind, 42
 - bind1st, 42
 - bind2nd, 42
- binders.h, 3440
- binomial_heap.hpp, 3441
- binomial_heap_base.hpp, 3441
- bitmap_allocator.h, 3442
 - _BALLOC_ALIGN_BYTES, 3443
- bitset, 3443, 3444
 - std::bitset, 2318, 2319
- bool_set, 3445
 - std::tr2::bool_set, 3187
- bool_set.tcc, 3446
- boolalpha
 - std, 576
 - std::basic_fstream, 1779
 - std::basic_ifstream, 1827
 - std::basic_ios, 1855
 - std::basic_iostream, 1915
 - std::basic_istream, 1961
 - std::basic_istream, 2012
 - std::basic_ofstream, 2052
 - std::basic_ostream, 2089
 - std::basic_ostream, 2130
 - std::basic_stringstream, 2294
 - std::ios_base, 2666
- Boolean Operations Classes, 43
- boost_concept_check.h, 3447
- Branch-Based, 44
- branch_policy.hpp, 3447
- bucket
 - __gnu_debug::_Safe_local_iterator, 955
- bucket_count
 - std::unordered_map, 3253
 - std::unordered_multimap, 3276
 - std::unordered_multiset, 3297
 - std::unordered_set, 3320
- c
 - std::queue, 3046
- c++0x_warning.h, 3448
- c++14_warning.h, 3448
- c++allocator.h, 3448
- c++config.h, 3448
- c++io.h, 3454
- c++locale.h, 3454
- c++locale_internal.h, 3455
- c_str
 - __gnu_cxx::__versa_string, 751
 - std::basic_string, 2181
- cache_line_size
 - __gnu_parallel::_Settings, 1104
- call_once
 - std, 576
 - std::once_flag, 2999
- capacity
 - __gnu_cxx::__versa_string, 751
 - __gnu_debug::basic_string, 995
 - std::basic_string, 2182
 - std::vector, 3347
- cassert, 3455
- cast.h, 3456
- category
 - std::locale, 2756
- cbefore_begin
 - std::forward_list, 2563
- cbegin
 - __gnu_cxx::__versa_string, 751
 - std, 576
 - std::basic_string, 2182
 - std::deque, 2493
 - std::forward_list, 2563
 - std::list, 2740
 - std::map, 2789
 - std::match_results, 2814
 - std::multimap, 2877
 - std::multiset, 2902
 - std::set, 3089
 - std::unordered_map, 3253
 - std::unordered_multimap, 3276

- std::unordered_multiset, [3297](#)
- std::unordered_set, [3320](#)
- std::vector, [3348](#)
- cc_hash_max_collision_check_resize_trigger
 - __gnu_pbds::cc_hash_max_collision_check_↵
 - resize_trigger, [1140](#)
- cc_hash_max_collision_check_resize_trigger_imp.hpp, [3456](#)
- cc_hash_table
 - __gnu_pbds::cc_hash_table, [1145–1147](#)
- cc_ht_map.hpp, [3456](#)
- ccomplex, [3457](#)
- cctype, [3457](#), [3458](#)
- cend
 - __gnu_cxx::__versa_string, [751](#)
 - std, [576](#)
 - std::basic_string, [2182](#)
 - std::deque, [2493](#)
 - std::forward_list, [2564](#)
 - std::list, [2740](#)
 - std::map, [2789](#)
 - std::match_results, [2814](#)
 - std::multimap, [2878](#)
 - std::multiset, [2902](#)
 - std::set, [3089](#)
 - std::unordered_map, [3253](#), [3254](#)
 - std::unordered_multimap, [3276](#)
 - std::unordered_multiset, [3298](#)
 - std::unordered_set, [3321](#)
 - std::vector, [3348](#)
- cerr
 - std, [646](#)
- cerrno, [3458](#)
- cfenv, [3458](#), [3459](#)
- cfloat, [3459](#)
- char_traits.h, [3460](#)
- char_type
 - std::__ctype_abstract_base, [1436](#)
 - std::basic_ios, [1836](#)
 - std::basic_streambuf, [2148](#)
 - std::collate, [2377](#)
 - std::collate_byname, [2383](#)
 - std::ctype< char >, [2414](#)
 - std::ctype< wchar_t >, [2428](#)
 - std::ctype_byname< char >, [2459](#)
 - std::istreambuf_iterator, [2716](#)
 - std::messages, [2832](#)
 - std::money_get, [2843](#)
 - std::money_put, [2848](#)
 - std::moneypunct, [2853](#)
 - std::num_get, [2933](#)
 - std::num_put, [2950](#)
 - std::numput, [2988](#)
 - std::ostream_iterator, [3000](#)
 - std::ostreambuf_iterator, [3003](#)
 - std::time_get, [3153](#)
 - std::time_put, [3175](#)
 - std::wbuffer_convert, [3364](#)
- checkers.h, [3460](#)
- chrono, [3460](#), [3463](#)
 - high_resolution_clock, [3463](#)
- cin
 - std, [646](#)
- cinttypes, [3464](#)
- ciso646, [3464](#)
- classic
 - std::locale, [2760](#)
- classic_table
 - std::ctype< char >, [2415](#)
 - std::ctype_byname< char >, [2459](#)
- clear
 - __gnu_cxx::__versa_string, [751](#)
 - std::basic_fstream, [1736](#)
 - std::basic_ifstream, [1794](#)
 - std::basic_ios, [1840](#)
 - std::basic_iostream, [1870](#)
 - std::basic_istream, [1929](#)
 - std::basic_istreamstring, [1978](#)
 - std::basic_ofstream, [2026](#)
 - std::basic_ostream, [2066](#)
 - std::basic_ostreamstring, [2106](#)
 - std::basic_string, [2182](#)
 - std::basic_stringstream, [2252](#)
 - std::deque, [2493](#)
 - std::experimental::fundamentals_v1::any, [2529](#)
 - std::forward_list, [2564](#)
 - std::list, [2740](#)
 - std::map, [2789](#)
 - std::multimap, [2878](#)
 - std::multiset, [2902](#)
 - std::set, [3089](#)
 - std::tr2::dynamic_bitset, [3195](#)
 - std::unordered_map, [3254](#)
 - std::unordered_multimap, [3277](#)
 - std::unordered_multiset, [3298](#)
 - std::unordered_set, [3321](#)
 - std::vector, [3348](#)
- climits, [3465](#)
- locale, [3465](#)
- clog
 - std, [646](#)
- close
 - __gnu_cxx::enc_filebuf, [817](#)
 - __gnu_cxx::stdio_filebuf, [875](#)
 - std::basic_filebuf, [1703](#)
 - std::basic_fstream, [1736](#)
 - std::basic_ifstream, [1794](#)
 - std::basic_ofstream, [2026](#)

- cmath, [3466](#), [3468](#)
- cmp_fn
 - __gnu_pbds::tree, [1372](#)
- cmp_fn_imps.hpp, [3471](#)
- code
 - std::regex_error, [3056](#)
- codecvt, [3471](#)
- codecvt.h, [3471](#)
- codecvt_specializations.h, [3472](#)
- collate
 - std::collate, [2377](#)
 - std::locale, [2764](#)
 - std::regex_constants, [700](#)
- combine
 - std::locale, [2760](#)
- comp_ellint_1
 - Mathematical Special Functions, [120](#), [141](#)
- comp_ellint_1f
 - Mathematical Special Functions, [121](#)
- comp_ellint_1l
 - Mathematical Special Functions, [121](#)
- comp_ellint_2
 - Mathematical Special Functions, [121](#), [141](#)
- comp_ellint_2f
 - Mathematical Special Functions, [122](#)
- comp_ellint_2l
 - Mathematical Special Functions, [122](#)
- comp_ellint_3
 - Mathematical Special Functions, [122](#), [141](#)
- comp_ellint_3f
 - Mathematical Special Functions, [123](#)
- comp_ellint_3l
 - Mathematical Special Functions, [123](#)
- compare
 - __gnu_cxx::__versa_string, [751](#)–[754](#)
 - __gnu_debug::basic_string, [995](#), [996](#)
 - std::basic_string, [2182](#)–[2185](#)
 - std::collate, [2378](#)
 - std::collate_byname, [2383](#)
 - std::sub_match, [3140](#), [3141](#)
- Comparison Classes, [45](#)
- compatibility.h, [3472](#)
- compiletime_settings.h, [3473](#)
 - _GLIBCXX_CALL, [3473](#)
 - _GLIBCXX_RANDOM_SHUFFLE_CONSIDER_L1, [3474](#)
 - _GLIBCXX_RANDOM_SHUFFLE_CONSIDER_T↔LB, [3474](#)
 - _GLIBCXX_SCALE_DOWN_FPU, [3474](#)
 - _GLIBCXX_VERBOSE_LEVEL, [3474](#)
- complex, [3474](#), [3478](#)
 - std::complex, [2388](#)
- Complex Numbers, [46](#)
 - abs, [49](#)
 - arg, [49](#)
 - conj, [49](#)
 - cos, [50](#)
 - cosh, [50](#)
 - exp, [50](#)
 - fabs, [50](#)
 - log, [50](#)
 - log10, [50](#)
 - norm, [50](#)
 - operator!=, [51](#)
 - operator<<, [54](#)
 - operator>>, [55](#)
 - operator*, [51](#)
 - operator*=: [51](#), [52](#)
 - operator+, [52](#)
 - operator+=, [52](#)
 - operator-, [52](#), [53](#)
 - operator-=, [53](#)
 - operator/, [53](#)
 - operator/=, [54](#)
 - operator=, [54](#)
 - operator==, [54](#), [55](#)
 - polar, [55](#)
 - pow, [55](#), [56](#)
 - sin, [56](#)
 - sinh, [56](#)
 - sqrt, [56](#)
 - tan, [56](#)
 - tanh, [56](#)
- complex.h, [3479](#)
- compose1
 - SGL, [305](#)
- compose2
 - SGL, [305](#)
- concept_check.h, [3480](#)
- concurrency.h, [3480](#)
- Concurrency, [57](#)
- cond_dealtor.hpp, [3481](#)
- cond_key_dtor_entry_dealtor.hpp, [3481](#)
- Condition Variables, [58](#)
 - cv_status, [58](#)
- condition_variable, [3481](#)
- conf_hyperg
 - __gnu_cxx, [375](#)
 - Mathematical Special Functions, [142](#)
- conf_hypergf
 - __gnu_cxx, [376](#)
- conf_hypergl
 - __gnu_cxx, [376](#)
- conj
 - Complex Numbers, [49](#)
- Const-propagating wrapper, [59](#)
- const_iterator
 - __gnu_pbds::trie_string_access_traits, [1386](#)

- std::set, 3084
- std::unordered_map, 3246
- std::unordered_multimap, 3270
- std::unordered_multiset, 3292
- std::unordered_set, 3314
- const_iterator.hpp, 3482, 3483
- const_iterator_, 1409
 - const_iterator_, 1411
 - const_pointer, 1410
 - const_reference, 1410
 - difference_type, 1410
 - iterator_category, 1410
 - m_p_tbl, 1412
 - operator!=, 1411
 - operator*, 1411
 - operator++, 1412
 - operator->, 1412
 - operator==, 1412
 - pointer, 1410
 - reference, 1411
 - value_type, 1411
- const_local_iterator
 - std::unordered_map, 3246
 - std::unordered_multimap, 3270
 - std::unordered_multiset, 3292
 - std::unordered_set, 3314
- const_pointer
 - __gnu_pbds::detail::binary_heap_const_iterator_, 1173
 - __gnu_pbds::detail::binary_heap_point_const_iterator_, 1177
 - __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_, 1223
 - __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_, 1227
 - const_iterator_, 1410
 - iterator_, 1414
 - point_const_iterator_, 1418
 - point_iterator_, 1421
 - std::allocator_traits, 1649
 - std::allocator_traits< allocator< _Tp > >, 1655
 - std::set, 3084
 - std::unordered_map, 3247
 - std::unordered_multimap, 3270
 - std::unordered_multiset, 3292
 - std::unordered_set, 3314
- const_pointer_cast
 - std, 577
- const_reference
 - __gnu_pbds::detail::bin_search_tree_const_node_it_, 1160
 - __gnu_pbds::detail::bin_search_tree_node_it_, 1166
 - __gnu_pbds::detail::binary_heap_const_iterator_, 1173
 - __gnu_pbds::detail::binary_heap_point_const_iterator_, 1177
 - __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_, 1223
 - __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_, 1227
 - const_iterator_, 1410
 - iterator_, 1414
 - point_const_iterator_, 1418
 - point_iterator_, 1421
 - std::allocator_traits, 1649
 - std::allocator_traits< allocator< _Tp > >, 1655
 - std::set, 3084
 - std::unordered_map, 3247
 - std::unordered_multimap, 3270
 - std::unordered_multiset, 3292
 - std::unordered_set, 3314
- const_reverse_iterator
 - std::set, 3084
- const_void_pointer
 - __gnu_cxx::__alloc_traits, 715
 - std::allocator_traits, 1650
 - std::allocator_traits< allocator< _Tp > >, 1655
- constant0
 - SGI, 305
- constant1
 - SGI, 305
- constant2
 - SGI, 305
- construct
 - __gnu_cxx::__alloc_traits, 717
 - std::allocator_traits, 1652
 - std::allocator_traits< allocator< _Tp > >, 1657
- constructor_destructor_fn_imps.hpp, 3483
- constructor_destructor_no_store_hash_fn_imps.hpp, 3483, 3484
- constructor_destructor_store_hash_fn_imps.hpp, 3484
- constructors_destructor_fn_imps.hpp, 3484–3486
- container_base_dispatch.hpp, 3486
- container_type
 - std::back_insert_iterator, 1691
 - std::front_insert_iterator, 2579
 - std::insert_iterator, 2648
- Containers, 61, 62
- converted
 - std::wstring_convert, 3386
- copy
 - __gnu_cxx::__versa_string, 755
 - Mutating, 150
 - std::basic_string, 2185
- copy_backward
 - Mutating, 150
- copy_exception
 - Exceptions, 74
- copy_if
 - Mutating, 150

- copy_n
 - Mutating, [151](#)
 - SGL, [305](#)
- copy_options
 - Filesystem, [82](#)
- copyfmt
 - std::basic_fstream, [1736](#)
 - std::basic_ifstream, [1794](#)
 - std::basic_ios, [1841](#)
 - std::basic_iostream, [1872](#)
 - std::basic_istream, [1929](#)
 - std::basic_istreamstream, [1979](#)
 - std::basic_ofstream, [2027](#)
 - std::basic_ostream, [2066](#)
 - std::basic_ostreamstream, [2106](#)
 - std::basic_stringstream, [2252](#)
- cos
 - Complex Numbers, [50](#)
- cosh
 - Complex Numbers, [50](#)
- count
 - Non-Mutating, [175](#)
 - std::bitset, [2320](#)
 - std::map, [2789](#), [2790](#)
 - std::multimap, [2878](#)
 - std::multiset, [2902](#)
 - std::set, [3090](#), [3091](#)
 - std::tr2::dynamic_bitset, [3195](#)
 - std::unordered_map, [3254](#)
 - std::unordered_multimap, [3277](#)
 - std::unordered_multiset, [3298](#)
 - std::unordered_set, [3321](#)
- count_if
 - Non-Mutating, [175](#)
- count_minimal_n
 - __gnu_parallel::Settings, [1104](#)
- cout
 - std, [646](#)
- cpp_type_traits.h, [3487](#)
- cpu_defines.h, [3487](#)
- crbegin
 - __gnu_cxx::__versa_string, [755](#)
 - std, [577](#)
 - std::basic_string, [2186](#)
 - std::deque, [2493](#)
 - std::list, [2741](#)
 - std::map, [2790](#)
 - std::multimap, [2879](#)
 - std::multiset, [2903](#)
 - std::set, [3091](#)
 - std::vector, [3348](#)
- cref
 - std, [577](#)
- cregex_token_iterator
 - Regular Expressions, [269](#)
- crend
 - __gnu_cxx::__versa_string, [755](#)
 - std, [577](#)
 - std::basic_string, [2186](#)
 - std::deque, [2493](#)
 - std::list, [2741](#)
 - std::map, [2790](#)
 - std::multimap, [2879](#)
 - std::multiset, [2903](#)
 - std::set, [3091](#)
 - std::vector, [3348](#)
- csetjmp, [3488](#)
- cshift
 - Numeric Arrays, [204](#)
- csignal, [3488](#)
- cstdalign, [3488](#)
- cstdarg, [3489](#)
- cstdbool, [3489](#), [3490](#)
- cstddef, [3490](#)
- cstdint, [3490](#), [3491](#)
- cstdio, [3491](#)
- cstdlib, [3492](#)
- cstring, [3492](#)
- csub_match
 - Regular Expressions, [269](#)
- ctgmach, [3493](#)
- ctime, [3494](#)
- ctype
 - std::ctype< char >, [2414](#), [2415](#)
 - std::ctype< wchar_t >, [2428](#), [2429](#)
 - std::locale, [2764](#)
- ctype_base.h, [3494](#)
- ctype_inline.h, [3495](#)
- cuchar, [3495](#)
- cur
 - std::basic_fstream, [1779](#)
 - std::basic_ifstream, [1828](#)
 - std::basic_ios, [1855](#)
 - std::basic_iostream, [1915](#)
 - std::basic_istream, [1962](#)
 - std::basic_istreamstream, [2012](#)
 - std::basic_ofstream, [2052](#)
 - std::basic_ostream, [2090](#)
 - std::basic_ostreamstream, [2130](#)
 - std::basic_stringstream, [2295](#)
 - std::ios_base, [2666](#)
- curr_symbol
 - std::moneypunct, [2855](#)
 - std::moneypunct_byname, [2863](#)
- current_exception
 - Exceptions, [74](#)
- cv_status
 - Condition Variables, [58](#)

- cwchar, [3495](#), [3496](#)
- cwctype, [3496](#), [3497](#)
- cxxabi.h, [3497](#)
 - __cxa_demangle, [3498](#)
- cxxabi_forced.h, [3499](#)
- cxxabi_tweaks.h, [3499](#)
- cyl_bessel_i
 - Mathematical Special Functions, [123](#), [142](#)
- cyl_bessel_if
 - Mathematical Special Functions, [124](#)
- cyl_bessel_il
 - Mathematical Special Functions, [124](#)
- cyl_bessel_j
 - Mathematical Special Functions, [124](#), [142](#)
- cyl_bessel_jf
 - Mathematical Special Functions, [125](#)
- cyl_bessel_jl
 - Mathematical Special Functions, [125](#)
- cyl_bessel_k
 - Mathematical Special Functions, [125](#), [142](#)
- cyl_bessel_kf
 - Mathematical Special Functions, [126](#)
- cyl_bessel_kl
 - Mathematical Special Functions, [126](#)
- cyl_neumann
 - Mathematical Special Functions, [126](#), [142](#)
- cyl_neumannf
 - Mathematical Special Functions, [127](#)
- cyl_neumannl
 - Mathematical Special Functions, [127](#)
- data
 - __gnu_cxx::__versa_string, [756](#)
 - std::basic_string, [2186](#)
 - std::vector, [3348](#)
- Data Structure Type, [63](#)
- date_order
 - std::time_get, [3154](#)
 - std::time_get_byname, [3165](#)
- deallocate
 - __gnu_cxx::__alloc_traits, [717](#)
 - std::allocator_traits, [1652](#)
 - std::allocator_traits< allocator< _Tp > >, [1657](#)
- debug.h, [3500](#)
- debug_allocator.h, [3501](#)
- debug_fn_imps.hpp, [3501](#)–[3503](#)
- debug_map_base.hpp, [3504](#)
- debug_no_store_hash_fn_imps.hpp, [3504](#)
- debug_store_hash_fn_imps.hpp, [3504](#)
- dec
 - std, [578](#)
 - std::basic_fstream, [1779](#)
 - std::basic_ifstream, [1828](#)
 - std::basic_ios, [1855](#)
 - std::basic_iostream, [1915](#)
 - std::basic_istream, [1962](#)
 - std::basic_istreamstream, [2012](#)
 - std::basic_ofstream, [2053](#)
 - std::basic_ostream, [2090](#)
 - std::basic_ostreamstream, [2130](#)
 - std::basic_stringstream, [2295](#)
 - std::ios_base, [2666](#)
- decimal, [3504](#)
- Decimal Floating-Point Arithmetic, [64](#)
- decimal128
 - std::decimal::decimal128, [2471](#)
- decimal32
 - std::decimal::decimal32, [2473](#)
- decimal32_to_long_long
 - std::decimal, [692](#)
- decimal64
 - std::decimal::decimal64, [2475](#)
- decimal_point
 - std::moneypunct, [2855](#)
 - std::moneypunct_byname, [2863](#)
 - std::numpunct, [2989](#)
 - std::numpunct_byname, [2994](#)
- default_delete
 - std::default_delete, [2476](#)
 - std::default_delete< _Tp[]>, [2477](#)
- defaultfloat
 - std, [578](#)
- defer_lock
 - Mutexes, [169](#)
- denorm_absent
 - std, [561](#)
- denorm_indeterminate
 - std, [561](#)
- denorm_min
 - std::numeric_limits, [2962](#)
- denorm_present
 - std, [561](#)
- densities
 - std::piecewise_constant_distribution, [3016](#)
 - std::piecewise_linear_distribution, [3021](#)
- deque, [3514](#), [3515](#)
 - std::deque, [2483](#)–[2485](#)
- deque.tcc, [3516](#)
- destroy
 - __gnu_cxx::__alloc_traits, [717](#)
 - std::allocator_traits, [1653](#)
 - std::allocator_traits< allocator< _Tp > >, [1658](#)
- Diagnostics, [65](#)
- difference_type
 - __gnu_pbds::detail::bin_search_tree_const_node_↵
it_, [1160](#)
 - __gnu_pbds::detail::bin_search_tree_node_it_, [1166](#)

- __gnu_pbds::detail::binary_heap_const_iterator_, 1174
- __gnu_pbds::detail::binary_heap_point_const_iterator_, 1177
- __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_, 1223
- __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_, 1228
- const_iterator_, 1410
- iterator_, 1414
- point_const_iterator_, 1419
- point_iterator_, 1422
- std::allocator_traits, 1650
- std::allocator_traits< allocator< _Tp > >, 1655
- std::back_insert_iterator, 1691
- std::front_insert_iterator, 2579
- std::insert_iterator, 2648
- std::istream_iterator, 2714
- std::istreambuf_iterator, 2716
- std::iterator, 2720
- std::ostream_iterator, 3000
- std::ostreambuf_iterator, 3003
- std::pointer_traits, 3029
- std::pointer_traits< _Tp * >, 3030
- std::raw_storage_iterator, 3052
- std::set, 3085
- std::unordered_map, 3247
- std::unordered_multimap, 3271
- std::unordered_multiset, 3292
- std::unordered_set, 3315
- digits
 - std::__numeric_limits_base, 1568
 - std::numeric_limits, 2964
- digits10
 - std::__numeric_limits_base, 1568
 - std::numeric_limits, 2964
- direct_mask_range_hashing_imp.hpp, 3517
- direct_mod_range_hashing_imp.hpp, 3517
- discard
 - std::discard_block_engine, 2506
 - std::independent_bits_engine, 2641
 - std::linear_congruential_engine, 2728
 - std::mersenne_twister_engine, 2827
 - std::shuffle_order_engine, 3126
 - std::subtract_with_carry_engine, 3145
- discard_block_engine
 - std::discard_block_engine, 2505, 2506
- distance
 - SGL, 306
 - std, 578
- do_compare
 - std::collate, 2378
 - std::collate_byname, 2383
- do_curr_symbol
 - std::moneypunct, 2855
 - std::moneypunct_byname, 2864
- do_date_order
 - std::time_get, 3154
 - std::time_get_byname, 3165
- do_decimal_point
 - std::moneypunct, 2855
 - std::moneypunct_byname, 2864
 - std::numpunct, 2989
 - std::numpunct_byname, 2994
- do_falsename
 - std::numpunct, 2989
 - std::numpunct_byname, 2995
- do_frac_digits
 - std::moneypunct, 2856
 - std::moneypunct_byname, 2864
- do_get
 - std::messages, 2833
 - std::messages_byname, 2836
 - std::money_get, 2844
 - std::num_get, 2934–2939
 - std::time_get, 3154
 - std::time_get_byname, 3165
- do_get_date
 - std::time_get, 3155
 - std::time_get_byname, 3166
- do_get_monthname
 - std::time_get, 3156
 - std::time_get_byname, 3167
- do_get_time
 - std::time_get, 3156
 - std::time_get_byname, 3167
- do_get_weekday
 - std::time_get, 3157
 - std::time_get_byname, 3168
- do_get_year
 - std::time_get, 3157
 - std::time_get_byname, 3168
- do_grouping
 - std::moneypunct, 2856
 - std::moneypunct_byname, 2865
 - std::numpunct, 2990
 - std::numpunct_byname, 2995
- do_hash
 - std::collate, 2379
 - std::collate_byname, 2384
- do_is
 - std::__ctype_abstract_base, 1436, 1437
 - std::ctype, 2400
 - std::ctype< wchar_t >, 2429
 - std::ctype_byname, 2445
- do_narrow
 - std::__ctype_abstract_base, 1437, 1438
 - std::ctype, 2401

- std::ctype< char >, 2415, 2416
- std::ctype< wchar_t >, 2430
- std::ctype_byname, 2445, 2446
- std::ctype_byname< char >, 2459, 2460
- do_neg_format
 - std::moneypunct, 2856
 - std::moneypunct_byname, 2865
- do_negative_sign
 - std::moneypunct, 2857
 - std::moneypunct_byname, 2865
- do_out
 - std::__codecvt_abstract_base, 1431
 - std::codecvt, 2345
 - std::codecvt< _InternT, _ExternT, encoding_state >, 2349
 - std::codecvt< char, char, mbstate_t >, 2354
 - std::codecvt< char16_t, char, mbstate_t >, 2358
 - std::codecvt< char32_t, char, mbstate_t >, 2362
 - std::codecvt< wchar_t, char, mbstate_t >, 2367
 - std::codecvt_byname, 2373
- do_pos_format
 - std::moneypunct, 2857
 - std::moneypunct_byname, 2866
- do_positive_sign
 - std::moneypunct, 2857
 - std::moneypunct_byname, 2866
- do_put
 - std::money_put, 2849
 - std::num_put, 2950–2954
 - std::time_put, 3176
 - std::time_put_byname, 3179
- do_scan_is
 - std::__ctype_abstract_base, 1438
 - std::ctype, 2402
 - std::ctype< wchar_t >, 2431
 - std::ctype_byname, 2447
- do_scan_not
 - std::__ctype_abstract_base, 1439
 - std::ctype, 2402
 - std::ctype< wchar_t >, 2432
 - std::ctype_byname, 2447
- do_thousands_sep
 - std::moneypunct, 2858
 - std::moneypunct_byname, 2866
 - std::numpunct, 2990
 - std::numpunct_byname, 2995
- do_tolower
 - std::__ctype_abstract_base, 1439, 1440
 - std::ctype, 2403
 - std::ctype< char >, 2416, 2417
 - std::ctype< wchar_t >, 2432, 2433
 - std::ctype_byname, 2448
 - std::ctype_byname< char >, 2460, 2461
- do_toupper
 - std::__ctype_abstract_base, 1440, 1441
 - std::ctype, 2404
 - std::ctype< char >, 2417, 2418
 - std::ctype< wchar_t >, 2433
 - std::ctype_byname, 2449
 - std::ctype_byname< char >, 2461, 2462
- do_transform
 - std::collate, 2379
 - std::collate_byname, 2385
- do_truename
 - std::numpunct, 2990
 - std::numpunct_byname, 2996
- do_widen
 - std::__ctype_abstract_base, 1441, 1442
 - std::ctype, 2405
 - std::ctype< char >, 2418
 - std::ctype< wchar_t >, 2434
 - std::ctype_byname, 2450
 - std::ctype_byname< char >, 2462
- duration_cast
 - std::chrono, 682
- Dynamic Bitset., 66
 - operator!=, 67
 - operator<<, 68
 - operator<=, 68
 - operator>, 68
 - operator>>, 68
 - operator>=, 68
 - operator^, 69
 - operator-, 67
 - operator&, 67
 - operator|, 69
- dynamic_bitset, 3517
 - std::tr2::dynamic_bitset, 3193, 3194
- dynamic_bitset.tcc, 3518
- dynamic_pointer_cast
 - std, 579
- e_pos
 - __gnu_pbds::sample_trie_access_traits, 1365
 - __gnu_pbds::trie_string_access_traits, 1387
- e_type
 - __gnu_pbds::sample_trie_access_traits, 1365
 - __gnu_pbds::trie_string_access_traits, 1387
- ECMAScript
 - std::regex_constants, 701
- eback
 - __gnu_cxx::enc_filebuf, 817
 - __gnu_cxx::stdio_filebuf, 875
 - __gnu_cxx::stdio_sync_filebuf, 898
 - std::basic_filebuf, 1703
 - std::basic_streambuf, 2149
 - std::basic_stringbuf, 2223
 - std::wbuffer_convert, 3365

- egptr
 - __gnu_cxx::enc_filebuf, [818](#)
 - __gnu_cxx::stdio_filebuf, [875](#)
 - __gnu_cxx::stdio_sync_filebuf, [898](#)
 - std::basic_filebuf, [1704](#)
 - std::basic_streambuf, [2149](#)
 - std::basic_stringbuf, [2223](#)
 - std::wbuffer_convert, [3365](#)
- egrep
 - std::regex_constants, [701](#)
- element_type
 - std::auto_ptr, [1685](#)
 - std::pointer_traits, [3029](#)
 - std::pointer_traits<_Tp * >, [3030](#)
- ellint_1
 - Mathematical Special Functions, [127](#), [142](#)
- ellint_1f
 - Mathematical Special Functions, [128](#)
- ellint_1l
 - Mathematical Special Functions, [128](#)
- ellint_2
 - Mathematical Special Functions, [128](#), [142](#)
- ellint_2f
 - Mathematical Special Functions, [129](#)
- ellint_2l
 - Mathematical Special Functions, [129](#)
- ellint_3
 - Mathematical Special Functions, [129](#), [143](#)
- ellint_3f
 - Mathematical Special Functions, [130](#)
- ellint_3l
 - Mathematical Special Functions, [130](#)
- emplace
 - std::deque, [2493](#)
 - std::list, [2741](#)
 - std::map, [2790](#)
 - std::multimap, [2879](#)
 - std::multiset, [2903](#)
 - std::set, [3091](#)
 - std::unordered_map, [3254](#)
 - std::unordered_multimap, [3277](#)
 - std::unordered_multiset, [3299](#)
 - std::unordered_set, [3322](#)
 - std::vector, [3348](#)
- emplace_after
 - std::forward_list, [2564](#)
- emplace_front
 - std::forward_list, [2564](#)
- emplace_hint
 - std::map, [2792](#)
 - std::multimap, [2879](#)
 - std::multiset, [2904](#)
 - std::set, [3092](#)
 - std::unordered_map, [3255](#)
 - std::unordered_multimap, [3278](#)
 - std::unordered_multiset, [3299](#)
 - std::unordered_set, [3322](#)
- empty
 - __gnu_cxx::__versa_string, [756](#)
 - __gnu_debug::basic_string, [996](#)
 - __gnu_pbds::detail::cc_ht_map, [1189](#)
 - __gnu_pbds::detail::gp_ht_map, [1215](#)
 - std::basic_string, [2187](#)
 - std::deque, [2494](#)
 - std::experimental::fundamentals_v1::any, [2529](#)
 - std::forward_list, [2565](#)
 - std::list, [2741](#)
 - std::map, [2792](#)
 - std::match_results, [2814](#)
 - std::multimap, [2880](#)
 - std::multiset, [2904](#)
 - std::priority_queue, [3039](#)
 - std::queue, [3044](#)
 - std::set, [3092](#)
 - std::stack, [3133](#)
 - std::tr2::dynamic_bitset, [3196](#)
 - std::unordered_map, [3256](#)
 - std::unordered_multimap, [3278](#)
 - std::unordered_multiset, [3300](#)
 - std::unordered_set, [3323](#)
 - std::vector, [3349](#)
- enable_special_members.h, [3519](#)
- enc_filebuf.h, [3519](#)
- end
 - __gnu_cxx::__versa_string, [756](#)
 - __gnu_cxx::temporary_buffer, [917](#)
 - __gnu_parallel::PseudoSequence, [1095](#)
 - __gnu_pbds::sample_trie_access_traits, [1365](#)
 - __gnu_pbds::trie_string_access_traits, [1387](#)
 - Numeric Arrays, [205](#)
 - std, [579](#), [580](#)
 - std::_Temporary_buffer, [1634](#)
 - std::basic_fstream, [1779](#)
 - std::basic_ifstream, [1828](#)
 - std::basic_ios, [1855](#)
 - std::basic_iostream, [1915](#)
 - std::basic_istream, [1962](#)
 - std::basic_istreambuf_iterator, [2012](#)
 - std::basic_ofstream, [2053](#)
 - std::basic_ostream, [2090](#)
 - std::basic_ostringstream, [2130](#)
 - std::basic_string, [2187](#)
 - std::basic_stringstream, [2295](#)
 - std::deque, [2494](#)
 - std::forward_list, [2565](#)
 - std::ios_base, [2666](#)
 - std::list, [2741](#), [2742](#)
 - std::map, [2793](#)

- [std::match_results](#), 2815
 - [std::multimap](#), 2880
 - [std::multiset](#), 2904
 - [std::set](#), 3092
 - [std::unordered_map](#), 3256
 - [std::unordered_multimap](#), 3278, 3279
 - [std::unordered_multiset](#), 3300
 - [std::unordered_set](#), 3323, 3324
 - [std::vector](#), 3349
- [endl](#)
 - [std](#), 580
- [ends](#)
 - [std](#), 580
- [entry_cmp.hpp](#), 3520
- [entry_list_fn_imps.hpp](#), 3520
- [entry_metadata_base.hpp](#), 3520
- [entry_pred.hpp](#), 3520
- [eof](#)
 - [std::basic_fstream](#), 1737
 - [std::basic_ifstream](#), 1795
 - [std::basic_ios](#), 1841
 - [std::basic_iostream](#), 1872
 - [std::basic_istream](#), 1930
 - [std::basic_istreamstream](#), 1979
 - [std::basic_ofstream](#), 2027
 - [std::basic_ostream](#), 2067
 - [std::basic_ostreamstream](#), 2106
 - [std::basic_stringstream](#), 2253
- [eofbit](#)
 - [std::basic_fstream](#), 1780
 - [std::basic_ifstream](#), 1828
 - [std::basic_ios](#), 1856
 - [std::basic_iostream](#), 1916
 - [std::basic_istream](#), 1962
 - [std::basic_istreamstream](#), 2012
 - [std::basic_ofstream](#), 2053
 - [std::basic_ostream](#), 2090
 - [std::basic_ostreamstream](#), 2130
 - [std::basic_stringstream](#), 2295
 - [std::ios_base](#), 2667
- [epptr](#)
 - [__gnu_cxx::enc_filebuf](#), 818
 - [__gnu_cxx::stdio_filebuf](#), 876
 - [__gnu_cxx::stdio_sync_filebuf](#), 899
 - [std::basic_filebuf](#), 1704
 - [std::basic_streambuf](#), 2150
 - [std::basic_stringbuf](#), 2224
 - [std::wbuffer_convert](#), 3365
- [epsilon](#)
 - [std::numeric_limits](#), 2962
- [eq_by_less.hpp](#), 3521
- [equal](#)
 - [Non-Mutating](#), 176, 177
 - [std::istreambuf_iterator](#), 2718
- [equal_range](#)
 - [Binary Search](#), 37, 38
 - [std::map](#), 2793–2795
 - [std::multimap](#), 2880–2882
 - [std::multiset](#), 2904–2906
 - [std::set](#), 3093, 3094
 - [std::unordered_map](#), 3257
 - [std::unordered_multimap](#), 3279, 3280
 - [std::unordered_multiset](#), 3301
 - [std::unordered_set](#), 3324
- [equally_split.h](#), 3521
- [equals](#)
 - [std::tr2::bool_set](#), 3187
- [erase](#)
 - [__gnu_cxx::__versa_string](#), 757, 758
 - [__gnu_debug::basic_string](#), 996, 997
 - [std::basic_string](#), 2187, 2188
 - [std::deque](#), 2494, 2495
 - [std::list](#), 2742
 - [std::map](#), 2795, 2796
 - [std::multimap](#), 2882, 2883
 - [std::multiset](#), 2906, 2907
 - [std::set](#), 3095, 3096
 - [std::unordered_map](#), 3258, 3259
 - [std::unordered_multimap](#), 3280, 3281
 - [std::unordered_multiset](#), 3302, 3303
 - [std::unordered_set](#), 3325, 3326
 - [std::vector](#), 3349, 3350
- [erase_after](#)
 - [std::forward_list](#), 2565, 2566
- [erase_can_throw](#)
 - [__gnu_pbds::container_traits](#), 1151
- [erase_fn_imps.hpp](#), 3522–3524
- [erase_if.h](#), 3524
- [erase_no_store_hash_fn_imps.hpp](#), 3524
- [erase_store_hash_fn_imps.hpp](#), 3525
- [error_backref](#)
 - [std::regex_constants](#), 695
- [error_badbrace](#)
 - [std::regex_constants](#), 695
- [error_badrepeat](#)
 - [std::regex_constants](#), 695
- [error_brace](#)
 - [std::regex_constants](#), 696
- [error_brack](#)
 - [std::regex_constants](#), 696
- [error_collate](#)
 - [std::regex_constants](#), 696
- [error_complexity](#)
 - [std::regex_constants](#), 696
- [error_constants.h](#), 3525
- [error_ctype](#)
 - [std::regex_constants](#), 696
- [error_escape](#)

- std::regex_constants, 696
- error_paren
 - std::regex_constants, 696
- error_range
 - std::regex_constants, 696
- error_space
 - std::regex_constants, 696
- error_stack
 - std::regex_constants, 696
- error_type
 - std::regex_constants, 695
- event
 - std::basic_fstream, 1734
 - std::basic_ifstream, 1792
 - std::basic_ios, 1839
 - std::basic_iostream, 1869
 - std::basic_istream, 1928
 - std::basic_istreamstream, 1977
 - std::basic_ofstream, 2024
 - std::basic_ostream, 2064
 - std::basic_ostreamstream, 2104
 - std::basic_stringstream, 2249
 - std::ios_base, 2658
- event_callback
 - std::basic_fstream, 1732
 - std::basic_ifstream, 1790
 - std::basic_ios, 1836
 - std::basic_iostream, 1867
 - std::basic_istream, 1926
 - std::basic_istreamstream, 1974
 - std::basic_ofstream, 2022
 - std::basic_ostream, 2062
 - std::basic_ostreamstream, 2101
 - std::basic_stringstream, 2247
 - std::ios_base, 2656
- exception, 3526
- exception.hpp, 3526
- exception_defines.h, 3527
- exception_ptr.h, 3527
- Exceptions, 71, 72
 - __verbose_terminate_handler, 74
 - copy_exception, 74
 - current_exception, 74
 - get_terminate, 74
 - get_unexpected, 75
 - make_exception_ptr, 75
 - rethrow_exception, 75
 - rethrow_if_nested, 75
 - set_terminate, 75
 - set_unexpected, 75
 - terminate, 75
 - terminate_handler, 74
 - throw_with_nested, 75
 - uncaught_exception, 75
 - uncaught_exceptions, 75
 - unexpected, 75
 - unexpected_handler, 74
- exceptions
 - std::basic_fstream, 1737
 - std::basic_ifstream, 1795
 - std::basic_ios, 1841, 1842
 - std::basic_iostream, 1872, 1873
 - std::basic_istream, 1930
 - std::basic_istreamstream, 1979, 1980
 - std::basic_ofstream, 2027, 2028
 - std::basic_ostream, 2067
 - std::basic_ostreamstream, 2107
 - std::basic_stringstream, 2253
- exchange
 - std, 581
- exp
 - Complex Numbers, 50
- Experimental, 76
- experimental/algorithm
 - __sample, 3403
 - sample, 3403
- experimental/bits/shared_ptr.h
 - get_deleter, 3737
- experimental/functional
 - is_bind_expression_v, 3556
 - is_placeholder_v, 3556
 - make_boyer_moore_horspool_searcher, 3555
 - make_boyer_moore_searcher, 3555
 - make_default_searcher, 3555
 - not_fn, 3556
- experimental/iterator
 - make_ostream_joiner, 3586
- expint
 - Mathematical Special Functions, 130, 143
- expintf
 - Mathematical Special Functions, 131
- expintl
 - Mathematical Special Functions, 131
- exponential_distribution
 - std::exponential_distribution, 2544
- extc++.h, 3528
- extended
 - std::regex_constants, 701
- Extensions, 77
- external_load_access
 - __gnu_pbds::cc_hash_max_collision_check_↔
 - resize_trigger, 1140
 - __gnu_pbds::hash_load_check_resize_trigger, 1328
- extptr_allocator.h, 3528
- fabs
 - Complex Numbers, 50
 - std, 581

- facet
 - std::locale::facet, [2767](#)
- fail
 - std::basic_fstream, [1738](#)
 - std::basic_ifstream, [1796](#)
 - std::basic_ios, [1842](#)
 - std::basic_istream, [1873](#)
 - std::basic_istream, [1931](#)
 - std::basic_istream, [1980](#)
 - std::basic_ofstream, [2028](#)
 - std::basic_ostream, [2068](#)
 - std::basic_ostream, [2108](#)
 - std::basic_stringstream, [2254](#)
- failbit
 - std::basic_fstream, [1780](#)
 - std::basic_ifstream, [1828](#)
 - std::basic_ios, [1856](#)
 - std::basic_istream, [1916](#)
 - std::basic_istream, [1962](#)
 - std::basic_istream, [2013](#)
 - std::basic_ofstream, [2053](#)
 - std::basic_ostream, [2090](#)
 - std::basic_ostream, [2131](#)
 - std::basic_stringstream, [2295](#)
 - std::ios_base, [2667](#)
- failed
 - std::ostreambuf_iterator, [3005](#)
- false_type
 - Metaprogramming, [147](#)
- falsename
 - std::numpunct, [2991](#)
 - std::numpunct_byname, [2996](#)
- fd
 - __gnu_cxx::stdio_filebuf, [876](#)
- features.h, [3529](#)
 - _GLIBCXX_BAL_QUICKSORT, [3529](#)
 - _GLIBCXX_FIND_CONSTANT_SIZE_BLOCKS, [3529](#)
 - _GLIBCXX_FIND_EQUAL_SPLIT, [3530](#)
 - _GLIBCXX_FIND_GROWING_BLOCKS, [3530](#)
 - _GLIBCXX_MERGESORT, [3530](#)
 - _GLIBCXX_QUICKSORT, [3530](#)
 - _GLIBCXX_TREE_DYNAMIC_BALANCING, [3530](#)
 - _GLIBCXX_TREE_FULL_COPY, [3531](#)
 - _GLIBCXX_TREE_INITIAL_SPLITTING, [3531](#)
- fenv.h, [3531](#)
- file
 - __gnu_cxx::stdio_filebuf, [876](#)
 - __gnu_cxx::stdio_sync_filebuf, [899](#)
- filebuf
 - I/O, [99](#)
- Filesystem, [78](#)
 - copy_options, [82](#)
 - perms, [82](#)
- filesystem, [3531](#)
- fill
 - Mutating, [151](#)
 - std::basic_fstream, [1738](#)
 - std::basic_ifstream, [1796](#)
 - std::basic_ios, [1843](#)
 - std::basic_istream, [1874](#)
 - std::basic_istream, [1931](#)
 - std::basic_istream, [1981](#)
 - std::basic_ofstream, [2029](#)
 - std::basic_ostream, [2068](#)
 - std::basic_ostream, [2108](#)
 - std::basic_stringstream, [2254](#)
- fill_minimal_n
 - __gnu_parallel::_Settings, [1104](#)
- fill_n
 - Mutating, [152](#)
- find
 - __gnu_cxx::__versa_string, [758](#), [759](#)
 - __gnu_debug::basic_string, [997](#)
 - Non-Mutating, [178](#)
 - std::basic_string, [2189](#), [2190](#)
 - std::map, [2797](#), [2798](#)
 - std::multimap, [2884](#), [2885](#)
 - std::multiset, [2908](#), [2909](#)
 - std::set, [3096](#), [3097](#)
 - std::unordered_map, [3259](#), [3260](#)
 - std::unordered_multimap, [3282](#)
 - std::unordered_multiset, [3303](#), [3304](#)
 - std::unordered_set, [3327](#)
- find.h, [3532](#)
- find_by_order
 - __gnu_pbds::tree_order_statistics_node_update, [1375](#)
 - __gnu_pbds::trie_order_statistics_node_update, [1381](#)
- find_end
 - Non-Mutating, [178](#), [179](#)
- find_first
 - std::tr2::dynamic_bitset, [3196](#)
- find_first_not_of
 - __gnu_cxx::__versa_string, [760](#), [761](#)
 - __gnu_debug::basic_string, [997](#)
 - std::basic_string, [2191](#), [2192](#)
- find_first_of
 - __gnu_cxx::__versa_string, [762](#), [763](#)
 - __gnu_debug::basic_string, [998](#)
 - Non-Mutating, [179](#), [180](#)
 - std::basic_string, [2192](#)–[2194](#)
- find_fn_imps.hpp, [3532](#)–[3534](#)
- find_if
 - Non-Mutating, [180](#)
- find_if_not
 - Non-Mutating, [181](#)

- find_increasing_factor
 - __gnu_parallel::Settings, 1104
- find_initial_block_size
 - __gnu_parallel::Settings, 1104
- find_last_not_of
 - __gnu_cxx::__versa_string, 764, 765
 - __gnu_debug::basic_string, 998
 - std::basic_string, 2194–2196
- find_last_of
 - __gnu_cxx::__versa_string, 766, 767
 - __gnu_debug::basic_string, 999
 - std::basic_string, 2196, 2197
- find_maximum_block_size
 - __gnu_parallel::Settings, 1104
- find_next
 - std::tr2::dynamic_bitset, 3196
- find_no_store_hash_fn_imps.hpp, 3534
- find_scale_factor
 - __gnu_parallel::Settings, 1104
- find_selectors.h, 3534
- find_sequential_search_size
 - __gnu_parallel::Settings, 1105
- find_store_hash_fn_imps.hpp, 3535
- first
 - __gnu_parallel::IteratorPair, 1059
 - std::pair, 3014
 - std::sub_match, 3142
- first_argument_type
 - __gnu_cxx::project1st, 852
 - __gnu_cxx::project2nd, 853
 - __gnu_parallel::EqualFromLess, 1053
 - __gnu_parallel::EqualTo, 1054
 - __gnu_parallel::Less, 1062
 - __gnu_parallel::Lexicographic, 1064
 - __gnu_parallel::LexicographicReverse, 1065
 - __gnu_parallel::Multiplies, 1088
 - __gnu_parallel::Plus, 1091
 - std::_Maybe_unary_or_binary_function<_Res, _T1, _T2>, 1626
 - std::binary_function, 2304
 - std::binary_negate, 2306
 - std::const_mem_fun1_ref_t, 2393
 - std::const_mem_fun1_t, 2394
 - std::divides, 2514
 - std::equal_to, 2518
 - std::experimental::fundamentals_v2::owner_less<shared_ptr<_Tp>>, 2540
 - std::experimental::fundamentals_v2::owner_less<weak_ptr<_Tp>>, 2541
 - std::greater, 2606
 - std::greater_equal, 2607
 - std::less, 2723
 - std::less_equal, 2725
 - std::logical_and, 2772
 - std::logical_or, 2776
 - std::mem_fun1_ref_t, 2820
 - std::mem_fun1_t, 2822
 - std::minus, 2837
 - std::modulus, 2839
 - std::multiplies, 2895
 - std::not_equal_to, 2929
 - std::owner_less<shared_ptr<_Tp>>, 3009
 - std::owner_less<weak_ptr<_Tp>>, 3010
 - std::plus, 3025
 - std::pointer_to_binary_function, 3027
- fixed
 - std, 581
 - std::basic_fstream, 1780
 - std::basic_ifstream, 1829
 - std::basic_ios, 1856
 - std::basic_iostream, 1916
 - std::basic_istream, 1963
 - std::basic_istreamstream, 2013
 - std::basic_ofstream, 2053
 - std::basic_ostream, 2091
 - std::basic_ostreamstream, 2131
 - std::basic_stringstream, 2296
 - std::ios_base, 2667
- flags
 - std::basic_fstream, 1739
 - std::basic_ifstream, 1797
 - std::basic_ios, 1843, 1844
 - std::basic_iostream, 1874, 1875
 - std::basic_istream, 1932
 - std::basic_istreamstream, 1981, 1982
 - std::basic_ofstream, 2029, 2030
 - std::basic_ostream, 2069
 - std::basic_ostreamstream, 2109
 - std::basic_regex, 2142
 - std::basic_stringstream, 2255
 - std::ios_base, 2659
- flip
 - std::bitset, 2320
 - std::tr2::dynamic_bitset, 3196, 3197
- float_denorm_style
 - std, 561
- float_round_style
 - std, 561
- floatfield
 - std::basic_fstream, 1780
 - std::basic_ifstream, 1829
 - std::basic_ios, 1856
 - std::basic_iostream, 1916
 - std::basic_istream, 1963
 - std::basic_istreamstream, 2013
 - std::basic_ofstream, 2054
 - std::basic_ostream, 2091
 - std::basic_ostreamstream, 2131

- std::basic_stringstream, 2296
- std::ios_base, 2667
- flush
 - std, 581
 - std::basic_fstream, 1740
 - std::basic_istream, 1875
 - std::basic_ofstream, 2030
 - std::basic_ostream, 2070
 - std::basic_ostringstream, 2109
 - std::basic_stringstream, 2256
- fmtflags
 - std::basic_fstream, 1732
 - std::basic_ifstream, 1790
 - std::basic_ios, 1837
 - std::basic_istream, 1867
 - std::basic_istream, 1926
 - std::basic_istream, 1975
 - std::basic_ofstream, 2022
 - std::basic_ostream, 2063
 - std::basic_ostringstream, 2102
 - std::basic_stringstream, 2247
 - std::ios_base, 2656
- for_each
 - Non-Mutating, 181
- for_each.h, 3535
- for_each_minimal_n
 - __gnu_parallel:: Settings, 1105
- for_each_selectors.h, 3536
- format
 - std::match_results, 2815
- format_default
 - std::regex_constants, 701
- format_first_only
 - std::regex_constants, 702
- format_no_copy
 - std::regex_constants, 702
- format_sed
 - std::regex_constants, 702
- formatter.h, 3536
- forward
 - Utilities, 353, 354
- forward_list, 3537–3539
 - std::forward_list, 2559–2561
- forward_list.h, 3540
- forward_list.tcc, 3541
- fpos
 - std::fpos, 2577
- frac_digits
 - std::moneypunct, 2858
 - std::moneypunct_byname, 2867
- from_bytes
 - std::wstring_convert, 3386, 3387
- front
 - __gnu_cxx::__versa_string, 768
- __gnu_debug::basic_string, 999
- std::basic_string, 2198
- std::deque, 2495
- std::forward_list, 2566
- std::list, 2743
- std::queue, 3045
- std::vector, 3350
- front_insert_iterator
 - std::front_insert_iterator, 2580
- front_inserter
 - Iterators, 109
- fs_dir.h, 3542
- fs_fwd.h, 3542
- fs_path.h, 3544
 - hash_value, 3545
 - operator!=, 3545
 - operator<, 3545
 - operator<<, 3546
 - operator<=, 3546
 - operator>, 3546
 - operator>>, 3546
 - operator>=, 3546
 - operator/, 3545
 - operator==, 3546
 - swap, 3546
 - u8path, 3547
- fstream, 3547
 - I/O, 99
- fstream.tcc, 3548
- functexcept.h, 3548
- function
 - std::function< _Res(_ArgTypes...)>, 2583, 2584
- Function Objects, 83
 - mem_fn, 84
- functional, 3549, 3553, 3554
- functional_hash.h, 3556
- functions.h, 3557
- future, 3559
 - std::future, 2590
 - std::future< _Res & >, 2593
 - std::future< void >, 2595
- future_category
 - Futures, 87
- future_errc
 - Futures, 86
- future_status
 - Futures, 86
- Futures, 85
 - async, 87
 - future_category, 87
 - future_errc, 86
 - future_status, 86
 - launch, 87
 - make_error_code, 87

- make_error_condition, [87](#)
- swap, [87](#)
- gamma_distribution
 - std::gamma_distribution, [2598](#)
- gbump
 - __gnu_cxx::enc_filebuf, [818](#)
 - __gnu_cxx::stdio_filebuf, [876](#)
 - __gnu_cxx::stdio_sync_filebuf, [899](#)
 - std::basic_filebuf, [1704](#)
 - std::basic_streambuf, [2150](#)
 - std::basic_stringbuf, [2224](#)
 - std::wbuffer_convert, [3365](#)
- gcount
 - std::basic_fstream, [1740](#)
 - std::basic_ifstream, [1798](#)
 - std::basic_iostream, [1875](#)
 - std::basic_istream, [1933](#)
 - std::basic_istreamstream, [1982](#)
 - std::basic_stringstream, [2256](#)
- generate
 - Mutating, [152](#)
- generate_canonical
 - Random Number Generation, [254](#)
- generate_minimal_n
 - __gnu_parallel::Settings, [1105](#)
- generate_n
 - Mutating, [153](#)
- get
 - __gnu_parallel::Settings, [1103](#)
 - std::__allocated_ptr, [1425](#)
 - std::auto_ptr, [1687](#)
 - std::basic_fstream, [1740–1743](#)
 - std::basic_ifstream, [1798–1800](#)
 - std::basic_iostream, [1876–1878](#)
 - std::basic_istream, [1933–1935](#)
 - std::basic_istreamstream, [1982–1985](#)
 - std::basic_stringstream, [2256–2259](#)
 - std::future, [2590](#)
 - std::future< _Res & >, [2593](#)
 - std::future< void >, [2595](#)
 - std::money_get, [2844, 2845](#)
 - std::num_get, [2940–2946](#)
 - std::shared_future, [3108](#)
 - std::shared_future< _Res & >, [3111](#)
 - std::time_get, [3158, 3159](#)
 - std::time_get_byname, [3169, 3170](#)
 - std::unique_ptr, [3236](#)
 - std::unique_ptr< _Tp[], _Dp >, [3241](#)
 - Utilities, [354, 355](#)
- get_actual_size
 - __gnu_pbds::hash_standard_resize_policy, [1332](#)
- get_allocator
 - __gnu_cxx::__versa_string, [768](#)
 - __gnu_debug::basic_string, [999](#)
 - std::basic_string, [2198](#)
 - std::deque, [2495](#)
 - std::forward_list, [2566](#)
 - std::list, [2743](#)
 - std::map, [2798](#)
 - std::match_results, [2816](#)
 - std::multimap, [2885](#)
 - std::multiset, [2909](#)
 - std::set, [3098](#)
 - std::tr2::dynamic_bitset, [3197](#)
 - std::unordered_map, [3260](#)
 - std::unordered_multimap, [3283](#)
 - std::unordered_multiset, [3304](#)
 - std::unordered_set, [3327](#)
- get_child
 - __gnu_pbds::detail::pat_trie_base::_Node_citer, [1262](#)
 - __gnu_pbds::detail::pat_trie_base::_Node_iter, [1265](#)
- get_comb_hash_fn
 - __gnu_pbds::detail::cc_ht_map, [1189](#)
- get_comb_probe_fn
 - __gnu_pbds::detail::gp_ht_map, [1215](#)
- get_date
 - std::time_get, [3159](#)
 - std::time_get_byname, [3170](#)
- get_deleter
 - experimental/bits/shared_ptr.h, [3737](#)
 - Pointer Abstractions, [244](#)
 - std::unique_ptr, [3236](#)
 - std::unique_ptr< _Tp[], _Dp >, [3241](#)
- get_eq_fn
 - __gnu_pbds::detail::cc_ht_map, [1189](#)
 - __gnu_pbds::detail::gp_ht_map, [1216](#)
- get_hash_fn
 - __gnu_pbds::detail::cc_ht_map, [1189, 1190](#)
 - __gnu_pbds::detail::gp_ht_map, [1216](#)
- get_id
 - std::this_thread, [707](#)
- get_l_child
 - __gnu_pbds::detail::bin_search_tree_const_node_↵
it_, [1162](#)
 - __gnu_pbds::detail::bin_search_tree_node_it_, [1167](#)
 - __gnu_pbds::detail::ov_tree_node_const_it_, [1241](#)
 - __gnu_pbds::detail::ov_tree_node_it_, [1243](#)
- get_load
 - __gnu_pbds::cc_hash_max_collision_check_↵
resize_trigger, [1141](#)
- get_loads
 - __gnu_pbds::hash_load_check_resize_trigger, [1328](#)
- get_metadata
 - __gnu_pbds::detail::bin_search_tree_const_node_↵
it_, [1162](#)
 - __gnu_pbds::detail::bin_search_tree_node_it_, [1167](#)

- __gnu_pbds::detail::pat_trie_base::_Node_citer, 1262
 - __gnu_pbds::detail::pat_trie_base::_Node_iter, 1265
- get_money
 - std, 581
- get_monthname
 - std::time_get, 3160
 - std::time_get_byname, 3171
- get_nearest_larger_size
 - __gnu_pbds::sample_size_policy, 1363
- get_nearest_smaller_size
 - __gnu_pbds::sample_size_policy, 1363
- get_new_handler
 - std, 582
- get_new_size
 - __gnu_pbds::hash_standard_resize_policy, 1332
 - __gnu_pbds::sample_resize_policy, 1358
- get_probe_fn
 - __gnu_pbds::detail::gp_ht_map, 1216
- get_r_child
 - __gnu_pbds::detail::bin_search_tree_const_node_iterator, 1162
 - __gnu_pbds::detail::bin_search_tree_node_iterator, 1167
 - __gnu_pbds::detail::ov_tree_node_const_iterator, 1241
 - __gnu_pbds::detail::ov_tree_node_iterator, 1243
- get_resize_policy
 - __gnu_pbds::detail::cc_ht_map, 1190
 - __gnu_pbds::detail::gp_ht_map, 1217
- get_size_policy
 - __gnu_pbds::hash_standard_resize_policy, 1332
- get_temporary_buffer
 - std, 582
- get_terminate
 - Exceptions, 74
- get_time
 - std, 582
 - std::time_get, 3160
 - std::time_get_byname, 3171
- get_trigger_policy
 - __gnu_pbds::hash_standard_resize_policy, 1333
- get_unexpected
 - Exceptions, 75
- get_weekday
 - std::time_get, 3161
 - std::time_get_byname, 3172
- get_year
 - std::time_get, 3161
 - std::time_get_byname, 3172
- getline
 - std, 583–585
 - std::basic_fstream, 1743, 1744
 - std::basic_ifstream, 1801, 1802
 - std::basic_iostream, 1879, 1880
 - std::basic_istream, 1936, 1937
 - std::basic_istream, 1986, 1987
 - std::basic_stringstream, 2259, 2260
- getloc
 - __gnu_cxx::enc_filebuf, 819
 - __gnu_cxx::stdio_filebuf, 877
 - __gnu_cxx::stdio_sync_filebuf, 900
 - std::basic_filebuf, 1705
 - std::basic_fstream, 1744
 - std::basic_ifstream, 1802
 - std::basic_ios, 1844
 - std::basic_iostream, 1880
 - std::basic_istream, 1937
 - std::basic_istream, 1987
 - std::basic_ofstream, 2030
 - std::basic_ostream, 2070
 - std::basic_ostringstream, 2110
 - std::basic_regex, 2142
 - std::basic_streambuf, 2151
 - std::basic_stringbuf, 2225
 - std::basic_stringstream, 2260
 - std::ios_base, 2660
 - std::regex_traits, 3065
 - std::wbuffer_convert, 3366
- global
 - std::locale, 2760
- good
 - std::basic_fstream, 1745
 - std::basic_ifstream, 1802
 - std::basic_ios, 1844
 - std::basic_iostream, 1880
 - std::basic_istream, 1937
 - std::basic_istream, 1987
 - std::basic_ofstream, 2031
 - std::basic_ostream, 2070
 - std::basic_ostringstream, 2110
 - std::basic_stringstream, 2261
- goodbit
 - std::basic_fstream, 1781
 - std::basic_ifstream, 1829
 - std::basic_ios, 1857
 - std::basic_iostream, 1917
 - std::basic_istream, 1963
 - std::basic_istream, 2013
 - std::basic_ofstream, 2054
 - std::basic_ostream, 2091
 - std::basic_ostringstream, 2131
 - std::basic_stringstream, 2296
 - std::ios_base, 2668
- gp_hash_table
 - __gnu_pbds::gp_hash_table, 1322–1324
- gp_ht_map.hpp, 3561
- gptr
 - __gnu_cxx::enc_filebuf, 819
 - __gnu_cxx::stdio_filebuf, 877

- `__gnu_cxx::stdio_sync_filebuf`, 900
 - `std::basic_filebuf`, 1705
 - `std::basic_streambuf`, 2151
 - `std::basic_stringbuf`, 2225
 - `std::wbuffer_convert`, 3366
- grep
 - `std::regex_constants`, 703
- grouping
 - `std::moneypunct`, 2858
 - `std::moneypunct_byname`, 2867
 - `std::numpunct`, 2991
 - `std::numpunct_byname`, 2996
- gslice
 - Numeric Arrays, 200, 201
- `gslice.h`, 3561
- `gslice_array`
 - Numeric Arrays, 201
- `gslice_array.h`, 3562
- has_denorm
 - `std::__numeric_limits_base`, 1569
 - `std::numeric_limits`, 2964
- has_denorm_loss
 - `std::__numeric_limits_base`, 1569
 - `std::numeric_limits`, 2964
- has_facet
 - Locales, 113
 - `std::locale`, 2763
 - `std::locale::id`, 2768
- has_infinity
 - `std::__numeric_limits_base`, 1569
 - `std::numeric_limits`, 2964
- has_quiet_NaN
 - `std::__numeric_limits_base`, 1569
 - `std::numeric_limits`, 2965
- has_signaling_NaN
 - `std::__numeric_limits_base`, 1569
 - `std::numeric_limits`, 2965
- hash
 - `std::collate`, 2380
 - `std::collate_byname`, 2385
- Hash-Based, 88
- `hash_bytes.h`, 3562
- `hash_eq_fn.hpp`, 3563
- `hash_exponential_size_policy`
 - `__gnu_pbds::hash_exponential_size_policy`, 1326
- `hash_exponential_size_policy_imp.hpp`, 3563
- `hash_fun.h`, 3563
- `hash_function`
 - `std::unordered_map`, 3260
 - `std::unordered_multimap`, 3283
 - `std::unordered_multiset`, 3304
 - `std::unordered_set`, 3328
- `hash_load_check_resize_trigger`
 - `__gnu_pbds::hash_load_check_resize_trigger`, 1328
- `hash_load_check_resize_trigger_imp.hpp`, 3563
- `hash_load_check_resize_trigger_size_base.hpp`, 3564
- `hash_map`, 3564
- `hash_policy.hpp`, 3565
- `hash_prime_size_policy`
 - `__gnu_pbds::hash_prime_size_policy`, 1330
- `hash_prime_size_policy_imp.hpp`, 3566
- `hash_set`, 3567
- `hash_standard_resize_policy`
 - `__gnu_pbds::hash_standard_resize_policy`, 1331, 1332
- `hash_standard_resize_policy_imp.hpp`, 3568
- `hash_value`
 - `fs_path.h`, 3545
- hasher
 - `std::unordered_map`, 3247
 - `std::unordered_multimap`, 3271
 - `std::unordered_multiset`, 3292
 - `std::unordered_set`, 3315
- Hashes, 89
- `hashtable.h`, 3568
- `hashtable_policy.h`, 3569
- Heap, 90
 - `is_heap`, 90, 91
 - `is_heap_until`, 91, 92
 - `make_heap`, 92
 - `pop_heap`, 93
 - `push_heap`, 93, 94
 - `sort_heap`, 94
- Heap-Based, 96
 - `priority_queue`, 97
- `helper_functions.h`, 3571
- hermite
 - Mathematical Special Functions, 131, 143
- hermitef
 - Mathematical Special Functions, 132
- hermitel
 - Mathematical Special Functions, 132
- hex
 - `std`, 585
 - `std::basic_fstream`, 1781
 - `std::basic_ifstream`, 1829
 - `std::basic_ios`, 1857
 - `std::basic_iostream`, 1917
 - `std::basic_istream`, 1963
 - `std::basic_istreamstream`, 2014
 - `std::basic_ofstream`, 2054
 - `std::basic_ostream`, 2091
 - `std::basic_ostreamstream`, 2132
 - `std::basic_stringstream`, 2296
 - `std::ios_base`, 2668
- hexfloat
 - `std`, 585

- high_resolution_clock
 - chrono, [3463](#)
- hours
 - std::chrono, [681](#)
- hyperg
 - __gnu_cxx, [376](#)
 - Mathematical Special Functions, [143](#)
- hypergf
 - __gnu_cxx, [378](#)
- hypergl
 - __gnu_cxx, [378](#)
- I/O, [98](#)
 - filebuf, [99](#)
 - fstream, [99](#)
 - ifstream, [99](#)
 - ios, [100](#)
 - iostream, [100](#)
 - istream, [100](#)
 - istringstream, [100](#)
 - ofstream, [100](#)
 - ostream, [100](#)
 - ostreamstream, [100](#)
 - streambuf, [100](#)
 - stringbuf, [101](#)
 - stringstream, [101](#)
 - wfilebuf, [101](#)
 - wfstream, [101](#)
 - wifstream, [101](#)
 - wios, [101](#)
 - wiostream, [101](#)
 - wistream, [101](#)
 - wistringstream, [102](#)
 - wofstream, [102](#)
 - wostream, [102](#)
 - wostringstream, [102](#)
 - wstreambuf, [102](#)
 - wstringbuf, [102](#)
 - wstringstream, [102](#)
- icase
 - std::regex_constants, [703](#)
- id
 - std::collate, [2381](#)
 - std::collate_byname, [2386](#)
 - std::ctype, [2412](#)
 - std::ctype< char >, [2425](#)
 - std::ctype< wchar_t >, [2441](#)
 - std::ctype_byname, [2456](#)
 - std::ctype_byname< char >, [2469](#)
 - std::locale::id, [2768](#)
 - std::messages, [2833](#)
 - std::messages_byname, [2836](#)
 - std::money_get, [2846](#)
 - std::money_put, [2851](#)
 - std::moneypunct, [2861](#)
 - std::moneypunct_byname, [2870](#)
 - std::num_get, [2947](#)
 - std::num_put, [2961](#)
 - std::numpunct, [2992](#)
 - std::numpunct_byname, [2998](#)
 - std::time_get, [3162](#)
 - std::time_get_byname, [3173](#)
 - std::time_put, [3178](#)
 - std::time_put_byname, [3181](#)
- identity_element
 - SGI, [306](#)
- ifstream
 - I/O, [99](#)
- ignore
 - std::basic_fstream, [1745](#), [1746](#)
 - std::basic_ifstream, [1802](#), [1803](#)
 - std::basic_iostream, [1880](#), [1881](#)
 - std::basic_istream, [1937](#), [1938](#)
 - std::basic_istringstream, [1987](#), [1988](#)
 - std::basic_stringstream, [2261](#), [2262](#)
- imbue
 - __gnu_cxx::enc_filebuf, [819](#)
 - __gnu_cxx::stdio_filebuf, [877](#)
 - __gnu_cxx::stdio_sync_filebuf, [900](#)
 - std::basic_filebuf, [1705](#)
 - std::basic_fstream, [1746](#)
 - std::basic_ifstream, [1804](#)
 - std::basic_ios, [1845](#)
 - std::basic_iostream, [1882](#)
 - std::basic_istream, [1939](#)
 - std::basic_istringstream, [1989](#)
 - std::basic_ofstream, [2031](#)
 - std::basic_ostream, [2070](#)
 - std::basic_ostreamstream, [2110](#)
 - std::basic_regex, [2142](#)
 - std::basic_streambuf, [2151](#)
 - std::basic_stringbuf, [2225](#)
 - std::basic_stringstream, [2262](#)
 - std::ios_base, [2660](#)
 - std::regex_traits, [3065](#)
 - std::wbuffer_convert, [3366](#)
- in
 - std::__codecvt_abstract_base, [1431](#)
 - std::basic_fstream, [1781](#)
 - std::basic_ifstream, [1830](#)
 - std::basic_ios, [1857](#)
 - std::basic_iostream, [1917](#)
 - std::basic_istream, [1964](#)
 - std::basic_istringstream, [2014](#)
 - std::basic_ofstream, [2054](#)
 - std::basic_ostream, [2092](#)
 - std::basic_ostreamstream, [2132](#)
 - std::basic_stringstream, [2297](#)

- std::codecvt, [2345](#)
- std::codecvt< _InternT, _ExternT, encoding_state >, [2349](#)
- std::codecvt< char, char, mbstate_t >, [2354](#)
- std::codecvt< char16_t, char, mbstate_t >, [2358](#)
- std::codecvt< char32_t, char, mbstate_t >, [2362](#)
- std::codecvt< wchar_t, char, mbstate_t >, [2367](#)
- std::codecvt_byname, [2373](#)
- std::ios_base, [2668](#)
- in_avail
 - __gnu_cxx::enc_filebuf, [820](#)
 - __gnu_cxx::stdio_filebuf, [878](#)
 - __gnu_cxx::stdio_sync_filebuf, [901](#)
 - std::basic_filebuf, [1706](#)
 - std::basic_streambuf, [2152](#)
 - std::basic_stringbuf, [2226](#)
 - std::wbuffer_convert, [3367](#)
- in_place
 - Optional values, [230](#)
- includes
 - Set Operation, [311](#)
- increment
 - std::linear_congruential_engine, [2731](#)
- independent_bits_engine
 - std::independent_bits_engine, [2639](#), [2640](#)
- index_sequence
 - std, [560](#)
- index_sequence_for
 - std, [560](#)
- indirect_array
 - Numeric Arrays, [201](#)
- indirect_array.h, [3572](#)
- infinity
 - std::numeric_limits, [2963](#)
- info_fn_imps.hpp, [3572](#)–[3574](#)
- init
 - std::basic_fstream, [1747](#)
 - std::basic_ifstream, [1804](#)
 - std::basic_ios, [1845](#)
 - std::basic_iostream, [1882](#)
 - std::basic_istream, [1939](#)
 - std::basic_istreamstream, [1989](#)
 - std::basic_ofstream, [2032](#)
 - std::basic_ostream, [2071](#)
 - std::basic_ostreamstream, [2111](#)
 - std::basic_stringstream, [2263](#)
- initializer_list, [3574](#)
- inner_product
 - std, [585](#), [586](#)
- inplace_merge
 - Sorting, [318](#)
- insert
 - __gnu_cxx::__versa_string, [768](#)–[772](#), [774](#)
 - __gnu_debug::basic_string, [999](#)–[1002](#)
 - std::basic_string, [2198](#)–[2202](#)
 - std::deque, [2496](#)–[2498](#)
 - std::list, [2743](#)–[2745](#)
 - std::map, [2798](#), [2800](#), [2801](#)
 - std::multimap, [2885](#), [2887](#), [2888](#)
 - std::multiset, [2909](#)–[2911](#)
 - std::set, [3098](#), [3099](#)
 - std::unordered_map, [3260](#)–[3263](#)
 - std::unordered_multimap, [3283](#)–[3285](#)
 - std::unordered_multiset, [3304](#), [3306](#), [3307](#)
 - std::unordered_set, [3328](#)–[3330](#)
 - std::vector, [3351](#), [3352](#)
- insert_after
 - std::forward_list, [2567](#), [2568](#)
- insert_fn_imps.hpp, [3575](#)–[3577](#)
- insert_iterator
 - std::insert_iterator, [2649](#)
- insert_join_fn_imps.hpp, [3577](#)
- insert_no_store_hash_fn_imps.hpp, [3577](#)
- insert_store_hash_fn_imps.hpp, [3577](#)
- inserter
 - Iterators, [109](#)
- int_type
 - std::basic_ios, [1837](#)
 - std::basic_streambuf, [2148](#)
 - std::istreambuf_iterator, [2716](#)
 - std::wbuffer_convert, [3364](#)
- internal
 - std, [586](#)
 - std::basic_fstream, [1781](#)
 - std::basic_ifstream, [1830](#)
 - std::basic_ios, [1857](#)
 - std::basic_iostream, [1917](#)
 - std::basic_istream, [1964](#)
 - std::basic_istreamstream, [2014](#)
 - std::basic_ofstream, [2054](#)
 - std::basic_ostream, [2092](#)
 - std::basic_ostreamstream, [2132](#)
 - std::basic_stringstream, [2297](#)
 - std::ios_base, [2668](#)
- intervals
 - std::piecewise_constant_distribution, [3016](#)
 - std::piecewise_linear_distribution, [3021](#)
- intl
 - std::moneypunct, [2861](#)
- Invalidation Guarantees, [103](#)
- io_errc
 - std, [562](#)
- iomanip, [3577](#)
- ios, [3579](#)
 - I/O, [100](#)
- ios_base.h, [3579](#)
- iosfwd, [3581](#)
- iostate

- std::basic_fstream, 1733
- std::basic_ifstream, 1791
- std::basic_ios, 1838
- std::basic_iostream, 1868
- std::basic_istream, 1927
- std::basic_istreamstream, 1975
- std::basic_ofstream, 2023
- std::basic_ostream, 2063
- std::basic_ostreamstream, 2102
- std::basic_stringstream, 2248
- std::ios_base, 2657
- iostream, 3582
 - I/O, 100
- iota
 - std, 586
- is
 - std::__ctype_abstract_base, 1442, 1443
 - std::ctype, 2406
 - std::ctype< char >, 2419
 - std::ctype< wchar_t >, 2435
 - std::ctype_byname, 2451
 - std::ctype_byname< char >, 2463
- is_always_equal
 - __gnu_cxx::__alloc_traits, 715
 - std::allocator_traits, 1650
 - std::allocator_traits< allocator< _Tp > >, 1655
- is_bind_expression_v
 - experimental/functional, 3556
- is_bounded
 - std::__numeric_limits_base, 1569
 - std::numeric_limits, 2965
- is_emptyset
 - std::tr2::bool_set, 3187
- is_exact
 - std::__numeric_limits_base, 1569
 - std::numeric_limits, 2965
- is_grow_needed
 - __gnu_pbds::cc_hash_max_collision_check_↵, 1141
 - __gnu_pbds::sample_resize_trigger, 1361
- is_heap
 - Heap, 90, 91
- is_heap_until
 - Heap, 91, 92
- is_iec559
 - std::__numeric_limits_base, 1569
 - std::numeric_limits, 2965
- is_indeterminate
 - std::tr2::bool_set, 3187
- is_integer
 - std::__numeric_limits_base, 1570
 - std::numeric_limits, 2965
- is_modulo
 - std::__numeric_limits_base, 1570
- std::numeric_limits, 2965
- is_open
 - __gnu_cxx::enc_filebuf, 820
 - __gnu_cxx::stdio_filebuf, 878
 - std::basic_filebuf, 1706
 - std::basic_fstream, 1747
 - std::basic_ifstream, 1804
 - std::basic_ofstream, 2032
- is_partitioned
 - Mutating, 153
- is_permutation
 - Non-Mutating, 182, 183
- is_placeholder_v
 - experimental/functional, 3556
- is_resize_needed
 - __gnu_pbds::cc_hash_max_collision_check_↵, 1141
 - __gnu_pbds::sample_resize_policy, 1358
 - __gnu_pbds::sample_resize_trigger, 1361
- is_signed
 - std::__numeric_limits_base, 1570
 - std::numeric_limits, 2966
- is_singleton
 - std::tr2::bool_set, 3187
- is_sorted
 - Sorting, 319
- is_sorted_until
 - Sorting, 320
- is_specialized
 - std::__numeric_limits_base, 1570
 - std::numeric_limits, 2966
- isalnum
 - std, 588
- isalpha
 - std, 588
- isblank
 - std, 588
- isctrl
 - std, 588
- isctype
 - std::regex_traits, 3066
- isdigit
 - std, 588
- isgraph
 - std, 588
- islower
 - std, 589
- isprint
 - std, 589
- ispunct
 - std, 589
- isspace
 - std, 589
- istream, 3583

- I/O, 100
- istream.tcc, 3584
- istream_iterator
 - std::istream_iterator, 2714
- istream_type
 - std::istreambuf_iterator, 2717
- istreambuf_iterator
 - std::istreambuf_iterator, 2718
- istreamstring
 - I/O, 100
- isupper
 - std, 589
- isxdigit
 - std, 589
- iter_swap
 - Mutating, 154
- iter_type
 - std::money_get, 2843
 - std::money_put, 2848
 - std::num_get, 2933
 - std::num_put, 2950
 - std::time_get, 3153
 - std::time_put, 3175
- iterator, 3585, 3586
 - std::set, 3085
 - std::unordered_map, 3247
 - std::unordered_multimap, 3271
 - std::unordered_multiset, 3293
 - std::unordered_set, 3315
- Iterator Tags, 104
- iterator.h, 3586
- iterator.hpp, 3587
- iterator_, 1413
 - const_pointer, 1414
 - const_reference, 1414
 - difference_type, 1414
 - iterator_, 1415
 - iterator_category, 1414
 - m_p_tbl, 1417
 - operator const point_iterator_, 1415
 - operator point_iterator_, 1415
 - operator!=, 1415, 1416
 - operator*, 1416
 - operator++, 1416
 - operator->, 1416
 - operator==, 1416
 - pointer, 1414
 - reference, 1415
 - value_type, 1415
- iterator_category
 - __gnu_pbds::detail::bin_search_tree_const_node_↵
it_, 1161
 - __gnu_pbds::detail::bin_search_tree_node_it_, 1166
- __gnu_pbds::detail::binary_heap_const_iterator_, 1174
- __gnu_pbds::detail::binary_heap_point_const_↵
iterator_, 1177
- __gnu_pbds::detail::left_child_next_sibling_heap_↵
const_iterator_, 1223
- __gnu_pbds::detail::left_child_next_sibling_heap_↵
node_point_const_iterator_, 1228
- const_iterator_, 1410
- iterator_, 1414
- point_const_iterator_, 1419
- point_iterator_, 1422
- std::back_insert_iterator, 1691
- std::front_insert_iterator, 2580
- std::insert_iterator, 2648
- std::istream_iterator, 2714
- std::istreambuf_iterator, 2717
- std::iterator, 2720
- std::ostream_iterator, 3000
- std::ostreambuf_iterator, 3004
- std::raw_storage_iterator, 3052
- std::reverse_iterator, 3072
- iterator_fn_imps.hpp, 3587
- iterator_tracker.h, 3587
- Iterators, 105
 - __iterator_category, 108
 - back_inserter, 108
 - front_inserter, 109
 - inserter, 109
 - make_reverse_iterator, 110
 - operator!=, 110
 - operator==, 110
- iterators_fn_imps.hpp, 3589, 3590
- iword
 - std::basic_fstream, 1747
 - std::basic_ifstream, 1805
 - std::basic_ios, 1845
 - std::basic_iostream, 1882
 - std::basic_istream, 1939
 - std::basic_istreamstring, 1989
 - std::basic_ofstream, 2032
 - std::basic_ostream, 2071
 - std::basic_ostringstream, 2111
 - std::basic_stringstream, 2263
 - std::ios_base, 2660
- k
 - std::negative_binomial_distribution, 2920
- key_comp
 - std::map, 2801
 - std::multimap, 2888
 - std::multiset, 2911
 - std::set, 3100
- key_compare

- std::set, [3085](#)
- key_eq
 - std::unordered_map, [3263](#)
 - std::unordered_multimap, [3286](#)
 - std::unordered_multiset, [3308](#)
 - std::unordered_set, [3330](#)
- key_equal
 - std::unordered_map, [3247](#)
 - std::unordered_multimap, [3271](#)
 - std::unordered_multiset, [3293](#)
 - std::unordered_set, [3315](#)
- key_type
 - std::set, [3085](#)
 - std::unordered_map, [3248](#)
 - std::unordered_multimap, [3271](#)
 - std::unordered_multiset, [3293](#)
 - std::unordered_set, [3315](#)
- kill_dependency
 - Atomics, [23](#)
- L1_cache_size
 - __gnu_parallel::Settings, [1105](#)
- L2_cache_size
 - __gnu_parallel::Settings, [1105](#)
- laguerre
 - Mathematical Special Functions, [132](#), [143](#)
- laguerref
 - Mathematical Special Functions, [133](#)
- laguerrel
 - Mathematical Special Functions, [133](#)
- lambda
 - std::exponential_distribution, [2545](#)
- launch
 - Futures, [87](#)
- left
 - std, [590](#)
 - std::basic_fstream, [1782](#)
 - std::basic_ifstream, [1830](#)
 - std::basic_ios, [1858](#)
 - std::basic_iostream, [1918](#)
 - std::basic_istream, [1964](#)
 - std::basic_istream, [2014](#)
 - std::basic_ofstream, [2055](#)
 - std::basic_ostream, [2092](#)
 - std::basic_ostream, [2132](#)
 - std::basic_stringstream, [2297](#)
 - std::ios_base, [2669](#)
- left_child_next_sibling_heap.hpp, [3590](#)
- left_child_next_sibling_heap_const_iterator_
 - __gnu_pbds::detail::left_child_next_sibling_heap_↔
const_iterator_, [1224](#)
- left_child_next_sibling_heap_node_point_const_iterator↔
 - __gnu_pbds::detail::left_child_next_sibling_heap_↔
node_point_const_iterator_, [1229](#)
- legendre
 - Mathematical Special Functions, [133](#), [143](#)
- legendref
 - Mathematical Special Functions, [134](#)
- legendrel
 - Mathematical Special Functions, [134](#)
- length
 - __gnu_cxx::__versa_string, [774](#)
 - __gnu_debug::basic_string, [1002](#)
 - std::basic_string, [2203](#)
 - std::match_results, [2816](#)
 - std::regex_traits, [3066](#)
 - std::sub_match, [3142](#)
- lexicographical_compare
 - Sorting, [320](#), [321](#)
- lexicographical_compare_3way
 - SGI, [306](#)
- lfts_config.h, [3590](#)
- limits, [3591](#)
- linear_congruential_engine
 - std::linear_congruential_engine, [2728](#)
- linear_probe_fn_imp.hpp, [3592](#)
- list, [3592–3594](#)
 - std::list, [2736–2738](#)
- List-Based, [111](#)
- list.tcc, [3595](#)
- list_partition
 - __gnu_parallel, [440](#)
- list_partition.h, [3595](#)
- list_update
 - __gnu_pbds::list_update, [1337](#)
- list_update_policy.hpp, [3595](#)
- load_factor
 - std::unordered_map, [3263](#)
 - std::unordered_multimap, [3286](#)
 - std::unordered_multiset, [3308](#)
 - std::unordered_set, [3331](#)
- local_iterator
 - std::unordered_map, [3248](#)
 - std::unordered_multimap, [3271](#)
 - std::unordered_multiset, [3293](#)
 - std::unordered_set, [3315](#)
- locale, [3596](#)
 - std::locale, [2756](#), [2758](#), [2759](#)
- locale_classes.h, [3596](#)
- locale_classes.tcc, [3596](#)
- locale_conv.h, [3597](#)
- locale_facets.h, [3598](#)
- locale_facets.tcc, [3599](#)
- locale_facets_nonio.h, [3600](#)
- locale_facets_nonio.tcc, [3601](#)
- localefwd.h, [3601](#)

- Locales, [112](#)
 - has_facet, [113](#)
 - use_facet, [113](#)
- lock
 - std, [590](#)
- log
 - Complex Numbers, [50](#)
- log10
 - Complex Numbers, [50](#)
- logic_error
 - std::logic_error, [2771](#)
- lookup_classname
 - std::regex_traits, [3067](#)
- lookup_collatename
 - std::regex_traits, [3068](#)
- losertree.h, [3602](#)
- lower_bound
 - Binary Search, [38](#), [39](#)
 - std::map, [2801](#)–[2803](#)
 - std::multimap, [2888](#), [2889](#)
 - std::multiset, [2911](#), [2912](#)
 - std::set, [3100](#), [3101](#)
- lowest
 - std::numeric_limits, [2963](#)
- lu_counter_metadata.hpp, [3603](#)
- lu_map.hpp, [3603](#)
- m_p_tbl
 - const_iterator_, [1412](#)
 - iterator_, [1417](#)
- macros.h, [3604](#)
 - _GLIBCXX_DEBUG_VERIFY_AT, [3607](#)
 - _glibcxx_check_erase, [3605](#)
 - _glibcxx_check_erase_after, [3605](#)
 - _glibcxx_check_erase_range, [3605](#)
 - _glibcxx_check_erase_range_after, [3605](#)
 - _glibcxx_check_heap_pred, [3605](#)
 - _glibcxx_check_insert, [3605](#)
 - _glibcxx_check_insert_after, [3605](#)
 - _glibcxx_check_insert_range, [3606](#)
 - _glibcxx_check_insert_range_after, [3606](#)
 - _glibcxx_check_partitioned_lower, [3606](#)
 - _glibcxx_check_partitioned_lower_pred, [3606](#)
 - _glibcxx_check_partitioned_upper_pred, [3606](#)
 - _glibcxx_check_sorted_pred, [3606](#)
- make_boyer_moore_horspool_searcher
 - experimental/functional, [3555](#)
- make_boyer_moore_searcher
 - experimental/functional, [3555](#)
- make_default_searcher
 - experimental/functional, [3555](#)
- make_error_code
 - Futures, [87](#)
- make_error_condition
 - Futures, [87](#)
- make_exception_ptr
 - Exceptions, [75](#)
- make_heap
 - Heap, [92](#)
- make_index_sequence
 - std, [560](#)
- make_integer_sequence
 - std, [560](#)
- make_ostream_joiner
 - experimental/iterator, [3586](#)
- make_pair
 - Utilities, [355](#)
- make_reverse_iterator
 - Iterators, [110](#)
- make_shared
 - Pointer Abstractions, [244](#)
- make_unique
 - Pointer Abstractions, [245](#)
- malloc_allocator.h, [3607](#)
- map, [3607](#), [3608](#)
 - std::map, [2785](#)–[2788](#)
- map.h, [3609](#), [3610](#)
- mapped_type
 - std::unordered_map, [3248](#)
 - std::unordered_multimap, [3272](#)
- mark_count
 - std::basic_regex, [2142](#)
- mask_array
 - Numeric Arrays, [201](#)
- mask_array.h, [3610](#)
- mask_based_range_hashing.hpp, [3611](#)
- match_any
 - std::regex_constants, [703](#)
- match_continuous
 - std::regex_constants, [703](#)
- match_default
 - std::regex_constants, [703](#)
- match_flag_type
 - std::regex_constants, [695](#)
- match_not_bol
 - std::regex_constants, [703](#)
- match_not_bow
 - std::regex_constants, [704](#)
- match_not_eol
 - std::regex_constants, [704](#)
- match_not_eow
 - std::regex_constants, [704](#)
- match_not_null
 - std::regex_constants, [704](#)
- match_prev_avail
 - std::regex_constants, [704](#)
- match_results
 - std::match_results, [2814](#)

- math.h, [3611](#)
- Mathematical Special Functions, [115](#), [139](#)
 - assoc_laguerre, [117](#), [141](#)
 - assoc_laguerref, [118](#)
 - assoc_laguerrel, [118](#)
 - assoc_legendre, [118](#), [141](#)
 - assoc_legendref, [119](#)
 - assoc_legendrel, [119](#)
 - beta, [119](#), [141](#)
 - betaf, [120](#)
 - betal, [120](#)
 - comp_ellint_1, [120](#), [141](#)
 - comp_ellint_1f, [121](#)
 - comp_ellint_1l, [121](#)
 - comp_ellint_2, [121](#), [141](#)
 - comp_ellint_2f, [122](#)
 - comp_ellint_2l, [122](#)
 - comp_ellint_3, [122](#), [141](#)
 - comp_ellint_3f, [123](#)
 - comp_ellint_3l, [123](#)
 - conf_hyperg, [142](#)
 - cyl_bessel_i, [123](#), [142](#)
 - cyl_bessel_if, [124](#)
 - cyl_bessel_il, [124](#)
 - cyl_bessel_j, [124](#), [142](#)
 - cyl_bessel_jf, [125](#)
 - cyl_bessel_jl, [125](#)
 - cyl_bessel_k, [125](#), [142](#)
 - cyl_bessel_kf, [126](#)
 - cyl_bessel_kl, [126](#)
 - cyl_neumann, [126](#), [142](#)
 - cyl_neumannf, [127](#)
 - cyl_neumannl, [127](#)
 - ellint_1, [127](#), [142](#)
 - ellint_1f, [128](#)
 - ellint_1l, [128](#)
 - ellint_2, [128](#), [142](#)
 - ellint_2f, [129](#)
 - ellint_2l, [129](#)
 - ellint_3, [129](#), [143](#)
 - ellint_3f, [130](#)
 - ellint_3l, [130](#)
 - expint, [130](#), [143](#)
 - expintf, [131](#)
 - expintl, [131](#)
 - hermite, [131](#), [143](#)
 - hermitef, [132](#)
 - hermitel, [132](#)
 - hyperg, [143](#)
 - laguerre, [132](#), [143](#)
 - laguerref, [133](#)
 - laguerrel, [133](#)
 - legendre, [133](#), [143](#)
 - legendref, [134](#)
 - legendrel, [134](#)
 - riemann_zeta, [134](#), [143](#)
 - riemann_zetaf, [135](#)
 - riemann_zetal, [135](#)
 - sph_bessel, [135](#), [144](#)
 - sph_besself, [136](#)
 - sph_bessell, [136](#)
 - sph_legendre, [136](#), [144](#)
 - sph_legendref, [137](#)
 - sph_legendrel, [137](#)
 - sph_neumann, [137](#), [144](#)
 - sph_neumannf, [138](#)
 - sph_neumannl, [138](#)
- max
 - __gnu_parallel, [440](#)
 - Numeric Arrays, [205](#)
 - Sorting, [321](#), [322](#)
 - std::bernoulli_distribution, [2301](#)
 - std::binomial_distribution, [2311](#)
 - std::cauchy_distribution, [2328](#)
 - std::chi_squared_distribution, [2336](#)
 - std::discard_block_engine, [2506](#)
 - std::discrete_distribution, [2511](#)
 - std::exponential_distribution, [2545](#)
 - std::extreme_value_distribution, [2548](#)
 - std::fisher_f_distribution, [2552](#)
 - std::gamma_distribution, [2598](#)
 - std::geometric_distribution, [2603](#)
 - std::independent_bits_engine, [2641](#)
 - std::linear_congruential_engine, [2728](#)
 - std::lognormal_distribution, [2778](#)
 - std::mersenne_twister_engine, [2828](#)
 - std::negative_binomial_distribution, [2920](#)
 - std::normal_distribution, [2926](#)
 - std::numeric_limits, [2963](#)
 - std::piecewise_constant_distribution, [3016](#)
 - std::piecewise_linear_distribution, [3021](#)
 - std::poisson_distribution, [3033](#)
 - std::shuffle_order_engine, [3127](#)
 - std::student_t_distribution, [3136](#)
 - std::subtract_with_carry_engine, [3145](#)
 - std::uniform_int_distribution, [3226](#)
 - std::uniform_real_distribution, [3229](#)
 - std::weibull_distribution, [3381](#)
- max_bucket_count
 - std::unordered_map, [3263](#)
 - std::unordered_multimap, [3286](#)
 - std::unordered_multiset, [3308](#)
 - std::unordered_set, [3331](#)
- max_count
 - __gnu_pbds::lu_counter_policy, [1340](#)
- max_digits10
 - std::__numeric_limits_base, [1570](#)
 - std::numeric_limits, [2966](#)

- max_element
 - Sorting, [322](#), [323](#)
- max_element_minimal_n
 - __gnu_parallel::Settings, [1105](#)
- max_exponent
 - std::numeric_limits_base, [1570](#)
 - std::numeric_limits, [2966](#)
- max_exponent10
 - std::numeric_limits_base, [1570](#)
 - std::numeric_limits, [2966](#)
- max_load_factor
 - std::unordered_map, [3263](#), [3264](#)
 - std::unordered_multimap, [3286](#)
 - std::unordered_multiset, [3308](#)
 - std::unordered_set, [3331](#)
- max_size
 - __gnu_cxx::__alloc_traits, [718](#)
 - __gnu_cxx::__versa_string, [775](#)
 - __gnu_debug::basic_string, [1002](#)
 - std::allocator_traits, [1653](#)
 - std::allocator_traits< allocator< _Tp > >, [1658](#)
 - std::basic_string, [2203](#)
 - std::deque, [2498](#)
 - std::forward_list, [2568](#)
 - std::list, [2745](#)
 - std::map, [2803](#)
 - std::match_results, [2816](#)
 - std::multimap, [2890](#)
 - std::multiset, [2913](#)
 - std::set, [3102](#)
 - std::tr2::dynamic_bitset, [3197](#)
 - std::unordered_map, [3264](#)
 - std::unordered_multimap, [3286](#)
 - std::unordered_multiset, [3309](#)
 - std::unordered_set, [3331](#)
 - std::vector, [3353](#)
- mean
 - std::normal_distribution, [2926](#)
 - std::poisson_distribution, [3033](#)
- mem_fn
 - Function Objects, [84](#)
- Memory, [145](#)
- memory, [3611](#)–[3613](#)
- memory_order
 - Atomics, [23](#)
- memory_resource, [3614](#)
- memoryfwd.h, [3615](#)
- merge
 - Sorting, [323](#), [324](#)
 - std::forward_list, [2569](#)
 - std::list, [2746](#)
- merge.h, [3615](#)
- merge_minimal_n
 - __gnu_parallel::Settings, [1105](#)
- merge_oversampling
 - __gnu_parallel::Settings, [1106](#)
- mersenne_twister_engine
 - std::mersenne_twister_engine, [2827](#)
- messages
 - std::locale, [2764](#)
 - std::messages, [2832](#)
- messages_members.h, [3616](#)
- metadata_const_reference
 - __gnu_pbds::detail::bin_search_tree_const_node_↔
it_, [1161](#)
 - __gnu_pbds::detail::bin_search_tree_node_it_, [1166](#)
- metadata_reference
 - __gnu_pbds::lu_counter_policy, [1339](#)
 - __gnu_pbds::lu_move_to_front_policy, [1341](#)
- metadata_type
 - __gnu_pbds::detail::bin_search_tree_const_node_↔
it_, [1161](#)
 - __gnu_pbds::detail::bin_search_tree_node_it_, [1166](#)
 - __gnu_pbds::detail::pat_trie_base::_Node_citer,
[1262](#)
 - __gnu_pbds::detail::pat_trie_base::_Node_iter, [1265](#)
 - __gnu_pbds::lu_counter_policy, [1339](#)
 - __gnu_pbds::lu_move_to_front_policy, [1341](#)
 - __gnu_pbds::sample_update_policy, [1367](#)
- Metaprogramming, [146](#)
 - false_type, [147](#)
 - true_type, [147](#)
- microseconds
 - std::chrono, [681](#)
- milliseconds
 - std::chrono, [682](#)
- min
 - __gnu_parallel, [440](#)
 - Numeric Arrays, [205](#)
 - Sorting, [324](#), [325](#)
 - std::bernoulli_distribution, [2301](#)
 - std::binomial_distribution, [2311](#)
 - std::cauchy_distribution, [2328](#)
 - std::chi_squared_distribution, [2336](#)
 - std::discard_block_engine, [2507](#)
 - std::discrete_distribution, [2511](#)
 - std::exponential_distribution, [2545](#)
 - std::extreme_value_distribution, [2549](#)
 - std::fisher_f_distribution, [2552](#)
 - std::gamma_distribution, [2599](#)
 - std::geometric_distribution, [2603](#)
 - std::independent_bits_engine, [2641](#)
 - std::linear_congruential_engine, [2729](#)
 - std::lognormal_distribution, [2778](#)
 - std::mersenne_twister_engine, [2828](#)
 - std::negative_binomial_distribution, [2920](#)
 - std::normal_distribution, [2926](#)
 - std::numeric_limits, [2963](#)

- `std::piecewise_constant_distribution`, 3016
 - `std::piecewise_linear_distribution`, 3021
 - `std::poisson_distribution`, 3033
 - `std::shuffle_order_engine`, 3127
 - `std::student_t_distribution`, 3136
 - `std::subtract_with_carry_engine`, 3145
 - `std::uniform_int_distribution`, 3226
 - `std::uniform_real_distribution`, 3230
 - `std::weibull_distribution`, 3381
- `min_element`
 - Sorting, 325, 326
- `min_element_minimal_n`
 - `__gnu_parallel::Settings`, 1106
- `min_exponent`
 - `std::__numeric_limits_base`, 1571
 - `std::numeric_limits`, 2966
- `min_exponent10`
 - `std::__numeric_limits_base`, 1571
 - `std::numeric_limits`, 2966
- `minmax`
 - Sorting, 326
- `minmax_element`
 - Sorting, 327
- `minstd_rand`
 - Random Number Generators, 256
- `minstd_rand0`
 - Random Number Generators, 256
- `minutes`
 - `std::chrono`, 682
- `mismatch`
 - Non-Mutating, 183–185
- `mod_based_range_hashing.hpp`, 3616
- `modulus`
 - `std::linear_congruential_engine`, 2731
- `monetary`
 - `std::locale`, 2764
- `money_get`
 - `std::money_get`, 2843
- `money_put`
 - `std::money_put`, 2848
- `moneypunct`
 - `std::moneypunct`, 2854
- `move`
 - Mutating, 154
 - Utilities, 356
- `move.h`, 3617
- `move_backward`
 - Mutating, 155
- `move_if_noexcept`
 - Utilities, 356
- `mt19937`
 - Random Number Generators, 256
- `mt19937_64`
 - Random Number Generators, 257
- `mt_allocator.h`, 3618
- `multimap`
 - `std::multimap`, 2874–2877
- `multimap.h`, 3618, 3619
- `multiplier`
 - `std::linear_congruential_engine`, 2731
- `multiseq_partition`
 - `__gnu_parallel`, 441
- `multiseq_selection`
 - `__gnu_parallel`, 441
- `multiseq_selection.h`, 3620
- `multiset`
 - `std::multiset`, 2899–2901
- `multiset.h`, 3621, 3622
- `multiway_merge`
 - `__gnu_parallel`, 442
- `multiway_merge.h`, 3623
 - `_GLIBCXX_PARALLEL_LENGTH`, 3626
- `multiway_merge_3_variant`
 - `__gnu_parallel`, 443
- `multiway_merge_4_variant`
 - `__gnu_parallel`, 444
- `multiway_merge_exact_splitting`
 - `__gnu_parallel`, 445
- `multiway_merge_loser_tree`
 - `__gnu_parallel`, 445
- `multiway_merge_loser_tree_sentinel`
 - `__gnu_parallel`, 445
- `multiway_merge_loser_tree_unguarded`
 - `__gnu_parallel`, 446
- `multiway_merge_minimal_k`
 - `__gnu_parallel::Settings`, 1106
- `multiway_merge_minimal_n`
 - `__gnu_parallel::Settings`, 1106
- `multiway_merge_oversampling`
 - `__gnu_parallel::Settings`, 1106
- `multiway_merge_sampling_splitting`
 - `__gnu_parallel`, 447
- `multiway_merge_sentinels`
 - `__gnu_parallel`, 447
- `multiway_mergesort.h`, 3626
- Mutating, 148
 - `copy`, 150
 - `copy_backward`, 150
 - `copy_if`, 150
 - `copy_n`, 151
 - `fill`, 151
 - `fill_n`, 152
 - `generate`, 152
 - `generate_n`, 153
 - `is_partitioned`, 153
 - `iter_swap`, 154
 - `move`, 154
 - `move_backward`, 155

- partition, [155](#)
- partition_copy, [156](#)
- partition_point, [156](#)
- random_shuffle, [157](#)
- remove, [157](#)
- remove_copy, [157](#)
- remove_copy_if, [158](#)
- remove_if, [158](#)
- replace, [159](#)
- replace_copy_if, [159](#)
- replace_if, [160](#)
- reverse, [160](#)
- reverse_copy, [161](#)
- rotate, [161](#)
- rotate_copy, [162](#)
- shuffle, [162](#)
- stable_partition, [163](#)
- swap_ranges, [163](#)
- transform, [164](#)
- unique, [165](#)
- unique_copy, [166](#)
- mutex, [3626](#)
- Mutexes, [168](#)
 - __cpp_lib_shared_timed_mutex, [169](#)
 - __shared_timed_mutex_base, [169](#)
 - adopt_lock, [169](#)
 - defer_lock, [169](#)
 - swap, [169](#)
 - try_to_lock, [169](#)
- name
 - std::locale, [2761](#)
 - std::type_info, [3221](#)
- nanoseconds
 - std::chrono, [682](#)
- narrow
 - std::__ctype_abstract_base, [1443](#), [1444](#)
 - std::basic_fstream, [1748](#)
 - std::basic_ifstream, [1805](#)
 - std::basic_ios, [1846](#)
 - std::basic_iostream, [1883](#)
 - std::basic_istream, [1940](#)
 - std::basic_istreamstream, [1990](#)
 - std::basic_ofstream, [2032](#)
 - std::basic_ostream, [2071](#)
 - std::basic_ostreamstream, [2111](#)
 - std::basic_stringstream, [2263](#)
 - std::ctype, [2407](#)
 - std::ctype< char >, [2420](#)
 - std::ctype< wchar_t >, [2436](#)
 - std::ctype_byname, [2452](#)
 - std::ctype_byname< char >, [2464](#)
- native_handle
 - std::thread, [3149](#)
- neg_format
 - std::moneypunct, [2859](#)
 - std::moneypunct_byname, [2867](#)
- negative_sign
 - std::moneypunct, [2859](#)
 - std::moneypunct_byname, [2868](#)
- Negators, [170](#)
 - not1, [171](#)
 - not2, [171](#)
- nested_exception.h, [3627](#)
- new, [3628](#)
 - operator delete, [3629](#)
 - operator delete[], [3630](#)
 - operator new, [3631](#)
 - operator new[], [3632](#)
- new_allocator.h, [3633](#)
- new_handler
 - std, [560](#)
- next_permutation
 - Sorting, [328](#)
- noboolalpha
 - std, [590](#)
- node.hpp, [3633](#), [3634](#)
- node_begin
 - __gnu_pbds::detail::ov_tree_map, [1238](#)
 - __gnu_pbds::detail::pat_trie_map, [1269](#)
 - __gnu_pbds::detail::rb_tree_map, [1280](#)
 - __gnu_pbds::detail::splay_tree_map, [1288](#)
- node_const_iterator
 - __gnu_pbds::detail::bin_search_tree_traits, [1169](#)
 - __gnu_pbds::detail::bin_search_tree_traits< Key, null_type, Cmp_Fn, Node_Update, Node, _Alloc >, [1170](#)
 - __gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >, [1299](#)
 - __gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, ov_tree_tag, _Alloc >, [1300](#)
 - __gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >, [1301](#)
 - __gnu_pbds::detail::tree_traits< Key, Mapped, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >, [1303](#)
 - __gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, splay_tree_tag, _Alloc >, [1304](#)
 - __gnu_pbds::detail::tree_traits< Key, null_type, Cmp_Fn, Node_Update, rb_tree_tag, _Alloc >, [1306](#)
 - __gnu_pbds::detail::trie_traits< Key, Mapped, _ATraits, Node_Update, pat_trie_tag, _Alloc >, [1310](#)

- __gnu_pbds::detail::trie_traits< Key, null_type, _↔
ATraits, Node_Update, pat_trie_tag, _Alloc >, 1312
- node_end
 - __gnu_pbds::detail::ov_tree_map, 1239
 - __gnu_pbds::detail::pat_trie_map, 1269
 - __gnu_pbds::detail::rb_tree_map, 1280
 - __gnu_pbds::detail::splay_tree_map, 1288
- node_iterators.hpp, 3634, 3635
- node_metadata_selector.hpp, 3635, 3636
- node_type
 - __gnu_pbds::detail::pat_trie_base, 1247
 - __gnu_pbds::detail::pat_trie_map, 1269
- node_update
 - __gnu_pbds::detail::trie_traits< Key, Mapped, _↔
ATraits, Node_Update, pat_trie_tag, _Alloc >, 1310
 - __gnu_pbds::detail::trie_traits< Key, null_type, _↔
ATraits, Node_Update, pat_trie_tag, _Alloc >, 1312
- Non-Mutating, 172
 - adjacent_find, 173, 174
 - all_of, 174
 - any_of, 175
 - count, 175
 - count_if, 175
 - equal, 176, 177
 - find, 178
 - find_end, 178, 179
 - find_first_of, 179, 180
 - find_if, 180
 - find_if_not, 181
 - for_each, 181
 - is_permutation, 182, 183
 - mismatch, 183–185
 - none_of, 185
 - search, 186
 - search_n, 187
- none
 - std::bitset, 2320
 - std::locale, 2765
 - std::tr2::dynamic_bitset, 3197
- none_of
 - Non-Mutating, 185
- norm
 - Complex Numbers, 50
- Normal Distributions, 189
 - operator!=, 190
 - operator<<, 190
 - operator>>, 191
- normal_distribution
 - std::normal_distribution, 2925
- noshowbase
 - std, 591
- noshowpoint
 - std, 591
- noshowpos
 - std, 591
- noskipws
 - std, 591
- nosubs
 - std::regex_constants, 704
- not1
 - Negators, 171
- not2
 - Negators, 171
- not_fn
 - experimental/functional, 3556
- notify_cleared
 - __gnu_pbds::cc_hash_max_collision_check_↔
resize_trigger, 1141
 - __gnu_pbds::hash_load_check_resize_trigger, 1328
 - __gnu_pbds::sample_resize_policy, 1358
 - __gnu_pbds::sample_resize_trigger, 1361
- notify_erase_search_collision
 - __gnu_pbds::cc_hash_max_collision_check_↔
resize_trigger, 1141
 - __gnu_pbds::sample_resize_policy, 1358
 - __gnu_pbds::sample_resize_trigger, 1361
- notify_erase_search_end
 - __gnu_pbds::cc_hash_max_collision_check_↔
resize_trigger, 1141
 - __gnu_pbds::sample_resize_policy, 1358
 - __gnu_pbds::sample_resize_trigger, 1361
- notify_erase_search_start
 - __gnu_pbds::cc_hash_max_collision_check_↔
resize_trigger, 1141
 - __gnu_pbds::sample_resize_policy, 1358
 - __gnu_pbds::sample_resize_trigger, 1361
- notify_erased
 - __gnu_pbds::cc_hash_max_collision_check_↔
resize_trigger, 1142
 - __gnu_pbds::sample_resize_policy, 1358
 - __gnu_pbds::sample_resize_trigger, 1361
- notify_externally_resized
 - __gnu_pbds::cc_hash_max_collision_check_↔
resize_trigger, 1142
 - __gnu_pbds::sample_resize_trigger, 1361
- notify_find_search_collision
 - __gnu_pbds::cc_hash_max_collision_check_↔
resize_trigger, 1142
 - __gnu_pbds::sample_resize_policy, 1358
 - __gnu_pbds::sample_resize_trigger, 1361
- notify_find_search_end
 - __gnu_pbds::cc_hash_max_collision_check_↔
resize_trigger, 1142
 - __gnu_pbds::sample_resize_policy, 1359
 - __gnu_pbds::sample_resize_trigger, 1361

- notify_find_search_start
 - __gnu_pbds::cc_hash_max_collision_check_↔
resize_trigger, [1142](#)
 - __gnu_pbds::sample_resize_policy, [1359](#)
 - __gnu_pbds::sample_resize_trigger, [1362](#)
- notify_insert_search_collision
 - __gnu_pbds::cc_hash_max_collision_check_↔
resize_trigger, [1142](#)
 - __gnu_pbds::sample_resize_policy, [1359](#)
 - __gnu_pbds::sample_resize_trigger, [1362](#)
- notify_insert_search_end
 - __gnu_pbds::cc_hash_max_collision_check_↔
resize_trigger, [1143](#)
 - __gnu_pbds::sample_resize_policy, [1359](#)
 - __gnu_pbds::sample_resize_trigger, [1362](#)
- notify_insert_search_start
 - __gnu_pbds::cc_hash_max_collision_check_↔
resize_trigger, [1143](#)
 - __gnu_pbds::sample_resize_policy, [1359](#)
 - __gnu_pbds::sample_resize_trigger, [1362](#)
- notify_inserted
 - __gnu_pbds::cc_hash_max_collision_check_↔
resize_trigger, [1143](#)
 - __gnu_pbds::hash_load_check_resize_trigger, [1328](#)
 - __gnu_pbds::sample_resize_policy, [1359](#)
 - __gnu_pbds::sample_resize_trigger, [1362](#)
- notify_resized
 - __gnu_pbds::cc_hash_max_collision_check_↔
resize_trigger, [1143](#)
 - __gnu_pbds::hash_load_check_resize_trigger, [1329](#)
 - __gnu_pbds::sample_range_hashing, [1355](#)
 - __gnu_pbds::sample_ranged_hash_fn, [1356](#)
 - __gnu_pbds::sample_resize_policy, [1359](#)
 - __gnu_pbds::sample_resize_trigger, [1362](#)
- nounitbuf
 - std, [591](#)
- nouppercase
 - std, [591](#)
- npos
 - __gnu_cxx::__versa_string, [793](#)
 - __gnu_debug::basic_string, [1009](#)
 - std::basic_string, [2219](#)
- nth_element
 - Sorting, [329](#)
- nth_element_minimal_n
 - __gnu_parallel::Settings, [1106](#)
- null_node_metadata.hpp, [3636](#)
- nullopt
 - Optional values, [230](#)
- num_blocks
 - std::tr2::dynamic_bitset, [3197](#)
- num_children
 - __gnu_pbds::detail::pat_trie_base::_Node_citer,
[1262](#)
- __gnu_pbds::detail::pat_trie_base::_Node_iter, [1265](#)
- num_get
 - std::num_get, [2933](#)
- num_put
 - std::num_put, [2950](#)
- numeric, [3637](#), [3639](#)
 - std::locale, [2765](#)
- Numeric Arrays, [192](#)
 - ~gslice, [203](#)
 - apply, [203](#), [204](#)
 - begin, [204](#)
 - cshift, [204](#)
 - end, [205](#)
 - gslice, [200](#), [201](#)
 - gslice_array, [201](#)
 - indirect_array, [201](#)
 - mask_array, [201](#)
 - max, [205](#)
 - min, [205](#)
 - operator!, [205](#)
 - operator<=, [211](#), [212](#)
 - operator>=, [216](#), [217](#)
 - operator*=, [207](#), [208](#)
 - operator^=, [221](#), [222](#)
 - operator+, [208](#)
 - operator+=, [208](#), [209](#)
 - operator-, [209](#)
 - operator=, [209](#), [210](#)
 - operator/=, [210](#), [211](#)
 - operator=, [212–216](#)
 - operator%=, [206](#)
 - operator&=, [206](#), [207](#)
 - operator[], [217–221](#)
 - operator|=, [222](#), [223](#)
 - operator~, [223](#)
 - resize, [223](#)
 - shift, [223](#)
 - size, [224](#)
 - slice, [201](#)
 - slice_array, [202](#)
 - start, [224](#)
 - stride, [225](#)
 - sum, [225](#)
 - swap, [225](#)
 - valarray, [202](#), [203](#)
- numeric_traits.h, [3640](#)
- numeric_fwd.h, [3641](#)
- Numerics, [226](#)
- numpunct
 - std::numpunct, [2988](#)
- oct
 - std, [592](#)
 - std::basic_fstream, [1782](#)

- std::basic_ifstream, 1830
- std::basic_ios, 1858
- std::basic_iostream, 1918
- std::basic_istream, 1964
- std::basic_istreamstream, 2015
- std::basic_ofstream, 2055
- std::basic_ostream, 2092
- std::basic_ostreamstream, 2133
- std::basic_stringstream, 2297
- std::ios_base, 2669
- off_type
 - std::basic_ios, 1838
 - std::basic_streambuf, 2148
 - std::wbuffer_convert, 3364
- ofstream
 - I/O, 100
- omp_loop.h, 3642
- omp_loop_static.h, 3643
- once_flag
 - std::once_flag, 2998
- open
 - __gnu_cxx::enc_filebuf, 820, 821
 - __gnu_cxx::stdio_filebuf, 878, 879
 - std::basic_filebuf, 1706, 1707
 - std::basic_fstream, 1748, 1749
 - std::basic_ifstream, 1806
 - std::basic_ofstream, 2033
- openmode
 - std::basic_fstream, 1733
 - std::basic_ifstream, 1791
 - std::basic_ios, 1838
 - std::basic_iostream, 1868
 - std::basic_istream, 1927
 - std::basic_istreamstream, 1976
 - std::basic_ofstream, 2023
 - std::basic_ostream, 2063
 - std::basic_ostreamstream, 2103
 - std::basic_stringstream, 2248
 - std::ios_base, 2657
- operator_iterator
 - __gnu_debug::_Safe_iterator, 941
 - __gnu_debug::_Safe_local_iterator, 955
- operator_RAlter
 - __gnu_parallel::_GuardedIterator, 1056
- operator bool
 - std::basic_fstream, 1749
 - std::basic_ifstream, 1806
 - std::basic_ios, 1846
 - std::basic_iostream, 1883
 - std::basic_istream, 1940
 - std::basic_istream::sentry, 1968
 - std::basic_istreamstream, 1990
 - std::basic_ofstream, 2034
 - std::basic_ostream, 2072
 - std::basic_ostream::sentry, 2096
 - std::basic_ostreamstream, 2112
 - std::basic_stringstream, 2264
 - std::function< _Res(_ArgTypes...)>, 2585
 - std::tr2::bool_set, 3188
 - std::unique_ptr, 3236
 - std::unique_ptr< _Tp[], _Dp >, 3241
- operator const point_iterator_
 - iterator_, 1415
- operator delete
 - new, 3629
- operator delete[]
 - new, 3630
- operator new
 - new, 3631
- operator new[]
 - new, 3632
- operator point_iterator_
 - iterator_, 1415
- operator streamoff
 - std::fpos, 2577
- operator string_type
 - std::sub_match, 3142
- operator!
 - Numeric Arrays, 205
 - std::basic_fstream, 1749
 - std::basic_ifstream, 1806
 - std::basic_ios, 1846
 - std::basic_iostream, 1884
 - std::basic_istream, 1941
 - std::basic_istreamstream, 1991
 - std::basic_ofstream, 2034
 - std::basic_ostream, 2072
 - std::basic_ostreamstream, 2112
 - std::basic_stringstream, 2264
- operator!=
 - __gnu_cxx, 378, 379
 - __gnu_pbds::detail::bin_search_tree_const_node_↔ it_, 1162
 - __gnu_pbds::detail::bin_search_tree_node_it_, 1167
 - __gnu_pbds::detail::binary_heap_const_iterator_, 1175
 - __gnu_pbds::detail::binary_heap_point_const_↔ iterator_, 1179
 - __gnu_pbds::detail::left_child_next_sibling_heap_↔ const_iterator_, 1224
 - __gnu_pbds::detail::left_child_next_sibling_heap_↔ node_point_const_iterator_, 1229
 - __gnu_pbds::detail::pat_trie_base::_Node_citer, 1262
 - __gnu_pbds::detail::pat_trie_base::_Node_iter, 1265
 - Bernoulli Distributions, 33
 - Complex Numbers, 51
 - const_iterator_, 1411

- Dynamic Bitset., 67
- fs_path.h, 3545
- iterator_, 1415, 1416
- Iterators, 110
- Normal Distributions, 190
- point_const_iterator_, 1420
- point_iterator_, 1423
- Poisson Distributions, 247, 248
- Random Number Generators, 257–259
- Regular Expressions, 270–273
- std, 592–594
- std::bitset, 2320
- std::locale, 2761
- std::regex_iterator, 3058
- std::regex_token_iterator, 3062
- std::rel_ops, 705
- Uniform Distributions, 346
- Utilities, 357
- operator<
 - __gnu_cxx, 381, 382
 - __gnu_parallel::GuardedIterator, 1056
 - fs_path.h, 3545
 - Regular Expressions, 273–275
 - std, 597–603
 - Utilities, 357
- operator<<
 - Bernoulli Distributions, 33, 34
 - Complex Numbers, 54
 - Dynamic Bitset., 68
 - fs_path.h, 3546
 - Normal Distributions, 190
 - Pointer Abstractions, 245
 - Poisson Distributions, 248, 249
 - Random Number Generators, 259
 - Regular Expressions, 275
 - std, 603–609
 - std::__detail, 655
 - std::basic_fstream, 1749–1755
 - std::basic_iostream, 1884–1887, 1889, 1890
 - std::basic_ofstream, 2034–2037, 2039–2041
 - std::basic_ostream, 2072–2078
 - std::basic_ostringstream, 2112–2118
 - std::basic_stringstream, 2264–2270
 - std::binomial_distribution, 2312
 - std::bitset, 2321
 - std::chi_squared_distribution, 2337
 - std::discard_block_engine, 2508
 - std::discrete_distribution, 2512
 - std::fisher_f_distribution, 2553
 - std::gamma_distribution, 2600
 - std::linear_congruential_engine, 2730
 - std::lognormal_distribution, 2779
 - std::mersenne_twister_engine, 2828
 - std::negative_binomial_distribution, 2922
 - std::normal_distribution, 2927
 - std::piecewise_constant_distribution, 3017
 - std::piecewise_linear_distribution, 3022
 - std::poisson_distribution, 3034
 - std::shuffle_order_engine, 3128
 - std::student_t_distribution, 3137
 - std::subtract_with_carry_engine, 3146
 - std::tr2::dynamic_bitset, 3198
 - Uniform Distributions, 347
- operator<=
 - Numeric Arrays, 211, 212
 - std::bitset, 2321
 - std::tr2::dynamic_bitset, 3199
- operator<=
 - __gnu_cxx, 383
 - __gnu_parallel::GuardedIterator, 1057
 - Dynamic Bitset., 68
 - fs_path.h, 3546
 - Regular Expressions, 276–278
 - std, 609–612
 - std::__debug, 652
 - std::__profile, 679
 - std::rel_ops, 705
 - Utilities, 357
- operator>
 - __gnu_cxx, 385, 386
 - Dynamic Bitset., 68
 - fs_path.h, 3546
 - Regular Expressions, 282–284
 - std, 618–621
 - std::__debug, 652
 - std::__profile, 679
 - std::rel_ops, 706
 - Utilities, 357
- operator>>
 - Bernoulli Distributions, 34
 - Complex Numbers, 55
 - Dynamic Bitset., 68
 - fs_path.h, 3546
 - Normal Distributions, 191
 - Poisson Distributions, 249, 250
 - std, 623–629
 - std::__detail, 656
 - std::basic_fstream, 1756–1758, 1760–1763
 - std::basic_ifstream, 1807–1809, 1811–1814
 - std::basic_iostream, 1892–1894, 1896–1899
 - std::basic_istream, 1941–1943, 1945–1948
 - std::basic_istream, 1991–1993, 1995–1998
 - std::basic_stringstream, 2271–2273, 2275–2278
 - std::binomial_distribution, 2313
 - std::bitset, 2321
 - std::chi_squared_distribution, 2337
 - std::discard_block_engine, 2508
 - std::discrete_distribution, 2512

- std::fisher_f_distribution, 2553
- std::gamma_distribution, 2600
- std::independent_bits_engine, 2642
- std::linear_congruential_engine, 2730
- std::lognormal_distribution, 2781
- std::mersenne_twister_engine, 2829
- std::negative_binomial_distribution, 2922
- std::normal_distribution, 2927
- std::piecewise_constant_distribution, 3018
- std::piecewise_linear_distribution, 3023
- std::poisson_distribution, 3035
- std::shuffle_order_engine, 3128
- std::student_t_distribution, 3137
- std::subtract_with_carry_engine, 3147
- std::tr2::dynamic_bitset, 3199
- Uniform Distributions, 347, 348
- operator>>=
 - Numeric Arrays, 216, 217
 - std::bitset, 2322
 - std::tr2::dynamic_bitset, 3199
- operator>=
 - __gnu_cxx, 386, 387
 - Dynamic Bitset., 68
 - fs_path.h, 3546
 - Regular Expressions, 285–287
 - std, 621–623
 - std::__debug, 652
 - std::__profile, 679
 - std::rel_ops, 706
 - Utilities, 358
- operator*
 - __gnu_debug::Safe_iterator, 941
 - __gnu_debug::Safe_local_iterator, 956
 - __gnu_parallel::GuardedIterator, 1056
 - __gnu_pbds::detail::bin_search_tree_const_node_↔
it_, 1162
 - __gnu_pbds::detail::bin_search_tree_node_it_, 1167
 - __gnu_pbds::detail::binary_heap_const_iterator_,
1175
 - __gnu_pbds::detail::binary_heap_point_const_↔
iterator_, 1179
 - __gnu_pbds::detail::left_child_next_sibling_heap_↔
const_iterator_, 1224
 - __gnu_pbds::detail::left_child_next_sibling_heap_↔
node_point_const_iterator_, 1229
 - __gnu_pbds::detail::ov_tree_node_it_, 1243
 - __gnu_pbds::detail::pat_trie_base::Node_citer,
1263
 - __gnu_pbds::detail::pat_trie_base::Node_iter, 1266
 - Complex Numbers, 51
 - const_iterator_, 1411
 - iterator_, 1416
 - point_const_iterator_, 1420
 - point_iterator_, 1423
 - std::auto_ptr, 1687
 - std::back_insert_iterator, 1692
 - std::front_insert_iterator, 2580
 - std::insert_iterator, 2649
 - std::istreambuf_iterator, 2718
 - std::ostreambuf_iterator, 3005
 - std::regex_iterator, 3058
 - std::regex_token_iterator, 3063
 - std::reverse_iterator, 3073
 - std::unique_ptr, 3236
- operator*=
 - Complex Numbers, 51, 52
 - Numeric Arrays, 207, 208
- operator^
 - Dynamic Bitset., 69
 - std, 630
 - std::regex_constants, 697, 698
- operator^=
 - Numeric Arrays, 221, 222
 - std::bitset, 2323
 - std::regex_constants, 698
 - std::tr2::dynamic_bitset, 3200
- operator()
 - __gnu_cxx::subtractive_rng, 915
 - __gnu_parallel::Nothing, 1089
 - __gnu_parallel::RandomNumber, 1099
 - __gnu_parallel::__accumulate_selector, 1011
 - __gnu_parallel::__adjacent_find_selector, 1013
 - __gnu_parallel::__count_if_selector, 1017
 - __gnu_parallel::__count_selector, 1019
 - __gnu_parallel::__fill_selector, 1020
 - __gnu_parallel::__find_first_of_selector, 1022
 - __gnu_parallel::__find_if_selector, 1024
 - __gnu_parallel::__for_each_selector, 1025
 - __gnu_parallel::__generate_selector, 1026
 - __gnu_parallel::__identity_selector, 1031
 - __gnu_parallel::__inner_product_selector, 1033
 - __gnu_parallel::__mismatch_selector, 1036
 - __gnu_parallel::__replace_if_selector, 1041
 - __gnu_parallel::__replace_selector, 1043
 - __gnu_parallel::__transform1_selector, 1044
 - __gnu_parallel::__transform2_selector, 1046
 - __gnu_pbds::direct_mask_range_hashing, 1318
 - __gnu_pbds::direct_mod_range_hashing, 1320
 - __gnu_pbds::linear_probe_fn, 1336
 - __gnu_pbds::lu_counter_policy, 1340
 - __gnu_pbds::lu_move_to_front_policy, 1341
 - __gnu_pbds::quadratic_probe_fn, 1349
 - __gnu_pbds::sample_probe_fn, 1353
 - __gnu_pbds::sample_range_hashing, 1355
 - __gnu_pbds::sample_ranged_hash_fn, 1356
 - __gnu_pbds::sample_trie_node_update, 1366
 - __gnu_pbds::sample_update_policy, 1367

- __gnu_pbds::tree_order_statistics_node_update, 1375
- __gnu_pbds::trie_order_statistics_node_update, 1381
- __gnu_pbds::trie_prefix_search_node_update, 1384
- std::bernoulli_distribution, 2301
- std::binomial_distribution, 2311
- std::cauchy_distribution, 2328
- std::chi_squared_distribution, 2336
- std::default_delete, 2476
- std::default_delete< _Tp[]>, 2477
- std::discard_block_engine, 2507
- std::discrete_distribution, 2511
- std::exponential_distribution, 2545
- std::extreme_value_distribution, 2549
- std::fisher_f_distribution, 2552
- std::function< _Res(_ArgTypes...)>, 2585
- std::gamma_distribution, 2599
- std::geometric_distribution, 2603
- std::independent_bits_engine, 2641
- std::linear_congruential_engine, 2729
- std::locale, 2761
- std::lognormal_distribution, 2779
- std::negative_binomial_distribution, 2921
- std::normal_distribution, 2926
- std::piecewise_constant_distribution, 3016
- std::piecewise_linear_distribution, 3021
- std::poisson_distribution, 3033
- std::shuffle_order_engine, 3127
- std::student_t_distribution, 3136
- std::subtract_with_carry_engine, 3145
- std::uniform_int_distribution, 3226
- std::uniform_real_distribution, 3230
- std::weibull_distribution, 3381
- operator+
 - __gnu_cxx, 379–381
 - Complex Numbers, 52
 - Numeric Arrays, 208
 - std, 595–597
 - std::fpos, 2577
 - std::reverse_iterator, 3074
- operator++
 - __gnu_debug::__Safe_iterator, 941, 942
 - __gnu_debug::__Safe_local_iterator, 956
 - __gnu_parallel::__GuardedIterator, 1056
 - const_iterator_, 1412
 - iterator_, 1416
 - std::back_insert_iterator, 1692
 - std::front_insert_iterator, 2580, 2581
 - std::insert_iterator, 2649
 - std::istreambuf_iterator, 2718, 2719
 - std::ostreambuf_iterator, 3005
 - std::regex_iterator, 3058
 - std::regex_token_iterator, 3063
 - std::reverse_iterator, 3074
- operator+=
 - __gnu_cxx::__versa_string, 775, 776
 - __gnu_debug::basic_string, 1002
 - Complex Numbers, 52
 - Numeric Arrays, 208, 209
 - std::basic_string, 2203, 2204
 - std::complex, 2388
 - std::fpos, 2577
 - std::reverse_iterator, 3074
- operator-
 - Complex Numbers, 52, 53
 - Dynamic Bitset., 67
 - Numeric Arrays, 209
 - std::fpos, 2577, 2578
 - std::reverse_iterator, 3075
- operator->
 - __gnu_debug::__Safe_iterator, 942
 - __gnu_debug::__Safe_local_iterator, 956
 - __gnu_pbds::detail::binary_heap_const_iterator_, 1175
 - __gnu_pbds::detail::binary_heap_point_const_iterator_, 1179
 - __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_, 1225
 - __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_, 1229
 - const_iterator_, 1412
 - iterator_, 1416
 - point_const_iterator_, 1420
 - point_iterator_, 1423
 - std::auto_ptr, 1687
 - std::regex_iterator, 3059
 - std::regex_token_iterator, 3063
 - std::reverse_iterator, 3076
 - std::unique_ptr, 3236
- operator--
 - __gnu_debug::__Safe_iterator, 942
 - std::reverse_iterator, 3075
- operator-=
 - Complex Numbers, 53
 - Numeric Arrays, 209, 210
 - std::complex, 2388
 - std::fpos, 2578
 - std::reverse_iterator, 3075
 - std::tr2::dynamic_bitset, 3198
- operator/
 - Complex Numbers, 53
 - fs_path.h, 3545
- operator/=
 - Complex Numbers, 54
 - Numeric Arrays, 210, 211
- operator=
 - __gnu_cxx::__versa_string, 776–778

- `__gnu_debug::Safe_iterator`, 943
- `__gnu_debug::Safe_local_iterator`, 957
- Complex Numbers, 54
- Numeric Arrays, 212–216
- `std::__allocated_ptr`, 1425
- `std::auto_ptr`, 1687, 1688
- `std::back_insert_iterator`, 1692
- `std::basic_regex`, 2142, 2143
- `std::basic_string`, 2205, 2206
- `std::deque`, 2498, 2499
- `std::experimental::fundamentals_v1::any`, 2530
- `std::forward_list`, 2569, 2570
- `std::front_insert_iterator`, 2581
- `std::function< _Res(_ArgTypes...)>`, 2585–2587
- `std::insert_iterator`, 2649
- `std::list`, 2746, 2747
- `std::locale`, 2762
- `std::map`, 2803, 2804
- `std::match_results`, 2816, 2817
- `std::multimap`, 2890, 2891
- `std::multiset`, 2913
- `std::once_flag`, 2999
- `std::ostream_iterator`, 3002
- `std::ostreambuf_iterator`, 3005
- `std::regex_iterator`, 3059
- `std::regex_token_iterator`, 3063
- `std::set`, 3102
- `std::tr2::dynamic_bitset`, 3199
- `std::unique_ptr`, 3236, 3237
- `std::unique_ptr< _Tp[], _Dp >`, 3241, 3242
- `std::unordered_map`, 3264
- `std::unordered_multimap`, 3287
- `std::unordered_multiset`, 3309
- `std::unordered_set`, 3332
- `std::vector`, 3353, 3354
- `operator==`
 - `__gnu_cxx`, 384, 385
 - `__gnu_pbds::detail::bin_search_tree_const_node_↔ it_`, 1162
 - `__gnu_pbds::detail::bin_search_tree_node_it_`, 1167
 - `__gnu_pbds::detail::binary_heap_const_iterator_`, 1175
 - `__gnu_pbds::detail::binary_heap_point_const_↔ iterator_`, 1179
 - `__gnu_pbds::detail::left_child_next_sibling_heap_↔ const_iterator_`, 1225
 - `__gnu_pbds::detail::left_child_next_sibling_heap_↔ node_point_const_iterator_`, 1229
 - `__gnu_pbds::detail::pat_trie_base::_Node_citer`, 1263
 - `__gnu_pbds::detail::pat_trie_base::_Node_iter`, 1266
 - Complex Numbers, 54, 55
 - `const_iterator_`, 1412
 - `fs_path.h`, 3546
 - `iterator_`, 1416
 - Iterators, 110
 - `point_const_iterator_`, 1420
 - `point_iterator_`, 1423
 - Regular Expressions, 278, 279, 281, 282
 - `std`, 612–618
 - `std::bernoulli_distribution`, 2302
 - `std::binomial_distribution`, 2313
 - `std::bitset`, 2321
 - `std::cauchy_distribution`, 2329
 - `std::chi_squared_distribution`, 2337
 - `std::discard_block_engine`, 2508
 - `std::discrete_distribution`, 2512
 - `std::exponential_distribution`, 2546
 - `std::extreme_value_distribution`, 2550
 - `std::fisher_f_distribution`, 2553
 - `std::gamma_distribution`, 2600
 - `std::geometric_distribution`, 2604
 - `std::independent_bits_engine`, 2642
 - `std::linear_congruential_engine`, 2730
 - `std::locale`, 2762
 - `std::lognormal_distribution`, 2781
 - `std::mersenne_twister_engine`, 2829
 - `std::negative_binomial_distribution`, 2922
 - `std::normal_distribution`, 2927
 - `std::piecewise_constant_distribution`, 3017
 - `std::piecewise_linear_distribution`, 3023
 - `std::poisson_distribution`, 3035
 - `std::regex_iterator`, 3059
 - `std::regex_token_iterator`, 3064
 - `std::shuffle_order_engine`, 3128
 - `std::student_t_distribution`, 3137
 - `std::subtract_with_carry_engine`, 3146
 - `std::uniform_int_distribution`, 3227
 - `std::uniform_real_distribution`, 3231
 - `std::weibull_distribution`, 3383
 - Utilities, 357
- `operator%=`
 - Numeric Arrays, 206
- `operator&`
 - Dynamic Bitset., 67
 - `std`, 595
 - `std::regex_constants`, 696, 697
- `operator&=`
 - Numeric Arrays, 206, 207
 - `std::bitset`, 2321
 - `std::regex_constants`, 697
 - `std::tr2::dynamic_bitset`, 3198
- `operator[]`
 - `__gnu_cxx::__versa_string`, 778, 779
 - Numeric Arrays, 217–221
 - `std::basic_string`, 2206, 2207
 - `std::bitset`, 2322
 - `std::deque`, 2500

- std::map, [2804](#)
- std::match_results, [2817](#)
- std::reverse_iterator, [3076](#)
- std::tr2::dynamic_bitset, [3200](#)
- std::unique_ptr< _Tp[], _Dp >, [3242](#)
- std::unordered_map, [3265](#)
- std::vector, [3354](#)
- operator |
 - Dynamic Bitset., [69](#)
 - std, [630](#)
 - std::regex_constants, [698](#), [699](#)
- operator | =
 - Numeric Arrays, [222](#), [223](#)
 - std::bitset, [2323](#)
 - std::regex_constants, [699](#)
 - std::tr2::dynamic_bitset, [3201](#)
- operator~
 - Numeric Arrays, [223](#)
 - std::bitset, [2323](#)
 - std::regex_constants, [699](#), [700](#)
 - std::tr2::dynamic_bitset, [3201](#)
- opt_random.h, [3643](#)
- optimize
 - std::regex_constants, [704](#)
- optional, [3643](#)
 - __constexpr_addressof, [3646](#)
- Optional values, [227](#)
 - __constexpr_addressof, [230](#)
 - in_place, [230](#)
 - nullopt, [230](#)
- order_of_key
 - __gnu_pbds::tree_order_statistics_node_update, [1376](#)
 - __gnu_pbds::trie_order_statistics_node_update, [1381](#)
- order_of_prefix
 - __gnu_pbds::trie_order_statistics_node_update, [1381](#)
- order_preserving
 - __gnu_pbds::container_traits, [1151](#)
- order_statistics_imp.hpp, [3646](#), [3647](#)
- ordered_base.h, [3647](#)
- os_defines.h, [3647](#)
- ostream, [3647](#)
 - I/O, [100](#)
- ostream.tcc, [3649](#)
- ostream_insert.h, [3649](#)
- ostream_iterator
 - std::ostream_iterator, [3001](#), [3002](#)
- ostream_type
 - std::ostream_iterator, [3000](#)
 - std::ostreambuf_iterator, [3004](#)
- ostreambuf_iterator
 - std::ostreambuf_iterator, [3005](#)
- ostreamstream
 - I/O, [100](#)
- out
 - std::__codecvt_abstract_base, [1432](#)
 - std::basic_fstream, [1782](#)
 - std::basic_ifstream, [1830](#)
 - std::basic_ios, [1858](#)
 - std::basic_iostream, [1918](#)
 - std::basic_istream, [1964](#)
 - std::basic_istreamstream, [2015](#)
 - std::basic_ofstream, [2055](#)
 - std::basic_ostream, [2092](#)
 - std::basic_ostreamstream, [2133](#)
 - std::basic_stringstream, [2297](#)
 - std::codecvt, [2346](#)
 - std::codecvt< _InternT, _ExternT, encoding_state >, [2350](#)
 - std::codecvt< char, char, mbstate_t >, [2355](#)
 - std::codecvt< char16_t, char, mbstate_t >, [2359](#)
 - std::codecvt< char32_t, char, mbstate_t >, [2363](#)
 - std::codecvt< wchar_t, char, mbstate_t >, [2368](#)
 - std::codecvt_byname, [2374](#)
 - std::ios_base, [2669](#)
- ov_tree_map.hpp, [3650](#)
- overflow
 - __gnu_cxx::enc_filebuf, [821](#)
 - __gnu_cxx::stdio_filebuf, [880](#)
 - __gnu_cxx::stdio_sync_filebuf, [901](#)
 - std::basic_filebuf, [1708](#)
 - std::basic_streambuf, [2152](#)
 - std::basic_stringbuf, [2226](#)
 - std::wbuffer_convert, [3367](#)
- p
 - std::bernoulli_distribution, [2301](#)
 - std::binomial_distribution, [2311](#)
 - std::geometric_distribution, [2603](#)
 - std::negative_binomial_distribution, [2921](#)
- pair
 - std::pair, [3014](#)
- pairing_heap.hpp, [3650](#)
- par_loop.h, [3651](#)
- parallel.h, [3651](#)
- parallel_balanced
 - __gnu_parallel, [410](#)
- parallel_multiway_merge
 - __gnu_parallel, [449](#)
- parallel_omp_loop
 - __gnu_parallel, [410](#)
- parallel_omp_loop_static
 - __gnu_parallel, [410](#)
- parallel_sort_mwms
 - __gnu_parallel, [449](#)
- parallel_sort_mwms_pu

- [__gnu_parallel, 450](#)
- [parallel_tag](#)
 - [__gnu_parallel::parallel_tag, 1127](#)
- [parallel_taskqueue](#)
 - [__gnu_parallel, 410](#)
- [parallel_unbalanced](#)
 - [__gnu_parallel, 410](#)
- [param](#)
 - [std::bernoulli_distribution, 2301](#)
 - [std::binomial_distribution, 2312](#)
 - [std::cauchy_distribution, 2328](#)
 - [std::chi_squared_distribution, 2336](#)
 - [std::discrete_distribution, 2511](#)
 - [std::exponential_distribution, 2545](#)
 - [std::extreme_value_distribution, 2549](#)
 - [std::fisher_f_distribution, 2552](#)
 - [std::gamma_distribution, 2599](#)
 - [std::geometric_distribution, 2603](#)
 - [std::lognormal_distribution, 2779](#)
 - [std::negative_binomial_distribution, 2921](#)
 - [std::normal_distribution, 2926](#)
 - [std::piecewise_constant_distribution, 3016, 3017](#)
 - [std::piecewise_linear_distribution, 3022](#)
 - [std::poisson_distribution, 3034](#)
 - [std::student_t_distribution, 3136](#)
 - [std::uniform_int_distribution, 3226](#)
 - [std::uniform_real_distribution, 3230](#)
 - [std::weibull_distribution, 3381, 3382](#)
- [parse_numbers.h, 3651](#)
- [partial_sort](#)
 - [Sorting, 330](#)
- [partial_sort_copy](#)
 - [Sorting, 331](#)
- [partial_sort_minimal_n](#)
 - [__gnu_parallel::_Settings, 1106](#)
- [partial_sum](#)
 - [std, 631](#)
- [partial_sum.h, 3652](#)
- [partial_sum_dilation](#)
 - [__gnu_parallel::_Settings, 1106](#)
- [partial_sum_minimal_n](#)
 - [__gnu_parallel::_Settings, 1107](#)
- [partition](#)
 - [Mutating, 155](#)
- [partition.h, 3652](#)
 - [_GLIBCXX_VOLATILE, 3653](#)
- [partition_chunk_share](#)
 - [__gnu_parallel::_Settings, 1107](#)
- [partition_chunk_size](#)
 - [__gnu_parallel::_Settings, 1107](#)
- [partition_copy](#)
 - [Mutating, 156](#)
- [partition_minimal_n](#)
 - [__gnu_parallel::_Settings, 1107](#)
- [partition_point](#)
 - [Mutating, 156](#)
- [pat_trie.hpp, 3653](#)
- [pat_trie_base.hpp, 3654](#)
- [pbackfail](#)
 - [__gnu_cxx::enc_filebuf, 822](#)
 - [__gnu_cxx::stdio_filebuf, 880](#)
 - [__gnu_cxx::stdio_sync_filebuf, 902](#)
 - [std::basic_filebuf, 1708](#)
 - [std::basic_streambuf, 2152](#)
 - [std::basic_stringbuf, 2227](#)
 - [std::wbuffer_convert, 3367](#)
- [pbase](#)
 - [__gnu_cxx::enc_filebuf, 822](#)
 - [__gnu_cxx::stdio_filebuf, 881](#)
 - [__gnu_cxx::stdio_sync_filebuf, 902](#)
 - [std::basic_filebuf, 1710](#)
 - [std::basic_streambuf, 2153](#)
 - [std::basic_stringbuf, 2227](#)
 - [std::wbuffer_convert, 3368](#)
- [pbump](#)
 - [__gnu_cxx::enc_filebuf, 823](#)
 - [__gnu_cxx::stdio_filebuf, 881](#)
 - [__gnu_cxx::stdio_sync_filebuf, 902](#)
 - [std::basic_filebuf, 1710](#)
 - [std::basic_streambuf, 2153](#)
 - [std::basic_stringbuf, 2228](#)
 - [std::wbuffer_convert, 3368](#)
- [peek](#)
 - [std::basic_fstream, 1763](#)
 - [std::basic_ifstream, 1814](#)
 - [std::basic_iostream, 1899](#)
 - [std::basic_istream, 1948](#)
 - [std::basic_istreamstream, 1998](#)
 - [std::basic_stringstream, 2278](#)
- [perms](#)
 - [Filesystem, 82](#)
- [piecewise_construct](#)
 - [Utilities, 359](#)
- [pod_char_traits.h, 3654](#)
- [point_const_iterator.hpp, 3655, 3656](#)
- [point_const_iterator_](#)
 - [1417](#)
 - [const_pointer, 1418](#)
 - [const_reference, 1418](#)
 - [difference_type, 1419](#)
 - [iterator_category, 1419](#)
 - [operator!=, 1420](#)
 - [operator*, 1420](#)
 - [operator->, 1420](#)
 - [operator==, 1420](#)
 - [point_const_iterator_, 1419](#)
 - [pointer, 1419](#)
 - [reference, 1419](#)
 - [value_type, 1419](#)

- point_iterator.hpp, 3656
- point_iterator_, 1421
 - const_pointer, 1421
 - const_reference, 1421
 - difference_type, 1422
 - iterator_category, 1422
 - operator!=, 1423
 - operator*, 1423
 - operator->, 1423
 - operator==, 1423
 - point_iterator_, 1422
 - pointer, 1422
 - reference, 1422
 - value_type, 1422
- point_iterators.hpp, 3656
- pointer
 - __gnu_pbds::detail::binary_heap_const_iterator_, 1174
 - __gnu_pbds::detail::binary_heap_point_const_iterator_, 1178
 - __gnu_pbds::detail::left_child_next_sibling_heap_const_iterator_, 1223
 - __gnu_pbds::detail::left_child_next_sibling_heap_node_point_const_iterator_, 1228
 - const_iterator_, 1410
 - iterator_, 1414
 - point_const_iterator_, 1419
 - point_iterator_, 1422
 - std::allocator_traits, 1650
 - std::allocator_traits< allocator< _Tp > >, 1656
 - std::back_insert_iterator, 1691
 - std::front_insert_iterator, 2580
 - std::insert_iterator, 2648
 - std::istream_iterator, 2714
 - std::istreambuf_iterator, 2717
 - std::iterator, 2720
 - std::ostream_iterator, 3001
 - std::ostreambuf_iterator, 3004
 - std::pointer_traits, 3029
 - std::pointer_traits< _Tp * >, 3031
 - std::raw_storage_iterator, 3052
 - std::set, 3085
 - std::unordered_map, 3248
 - std::unordered_multimap, 3272
 - std::unordered_multiset, 3293
 - std::unordered_set, 3316
- Pointer Abstractions, 231
 - allocate_shared, 235
 - atomic_compare_exchange_strong, 235
 - atomic_compare_exchange_strong_explicit, 236
 - atomic_compare_exchange_weak, 237
 - atomic_compare_exchange_weak_explicit, 238
 - atomic_exchange, 239
 - atomic_exchange_explicit, 240
 - atomic_is_lock_free, 241
 - atomic_load, 241, 242
 - atomic_load_explicit, 242
 - atomic_store, 243
 - atomic_store_explicit, 244
 - get_deleter, 244
 - make_shared, 244
 - make_unique, 245
 - operator<<, 245
- pointer.h, 3657
- pointer_to
 - std::pointer_traits< _Tp * >, 3031
- Poisson Distributions, 246
 - operator!=, 247, 248
 - operator<<, 248, 249
 - operator>>, 249, 250
- polar
 - Complex Numbers, 55
- Policy-Based Data Structures, 252
- policy_access_fn_imps.hpp, 3659, 3660
- pool_allocator.h, 3660
- pop
 - std::priority_queue, 3039
 - std::queue, 3045
 - std::stack, 3133
- pop_back
 - __gnu_cxx::__versa_string, 779
 - __gnu_parallel::RestrictedBoundedConcurrentQueue, 1100
 - std::basic_string, 2207
 - std::deque, 2500
 - std::list, 2748
 - std::vector, 3355
- pop_front
 - __gnu_parallel::RestrictedBoundedConcurrentQueue, 1100
 - std::deque, 2500
 - std::forward_list, 2570
 - std::list, 2748
- pop_heap
 - Heap, 93
- pos_format
 - std::moneypunct, 2860
 - std::moneypunct_byname, 2868
- pos_type
 - std::basic_ios, 1838
 - std::basic_streambuf, 2148
 - std::wbuffer_convert, 3364
- position
 - std::match_results, 2817
- positive_sign
 - std::moneypunct, 2860
 - std::moneypunct_byname, 2869
- postypes.h, 3660

- pow
 - Complex Numbers, [55](#), [56](#)
- power
 - SGI, [307](#)
- pptr
 - `__gnu_cxx::enc_filebuf`, [823](#)
 - `__gnu_cxx::stdio_filebuf`, [882](#)
 - `__gnu_cxx::stdio_sync_filebuf`, [903](#)
 - `std::basic_filebuf`, [1711](#)
 - `std::basic_streambuf`, [2154](#)
 - `std::basic_stringbuf`, [2228](#)
 - `std::wbuffer_convert`, [3369](#)
- precision
 - `std::basic_fstream`, [1763](#), [1764](#)
 - `std::basic_ifstream`, [1814](#), [1815](#)
 - `std::basic_ios`, [1847](#)
 - `std::basic_istream`, [1899](#), [1900](#)
 - `std::basic_istream`, [1948](#), [1949](#)
 - `std::basic_istream`, [1998](#), [1999](#)
 - `std::basic_ofstream`, [2041](#), [2042](#)
 - `std::basic_ostream`, [2079](#)
 - `std::basic_ostream`, [2119](#)
 - `std::basic_stringstream`, [2278](#), [2279](#)
 - `std::ios_base`, [2661](#)
- `predefined_ops.h`, [3661](#)
- prefix
 - `std::match_results`, [2817](#)
- prefix_range
 - `__gnu_pbds::trie_prefix_search_node_update`, [1384](#), [1385](#)
- `prefix_search_node_update_imp.hpp`, [3662](#)
- prev_permutation
 - Sorting, [332](#)
- priority_queue
 - Heap-Based, [97](#)
 - `std::priority_queue`, [3038](#)
- `priority_queue.hpp`, [3662](#)
- `priority_queue_base_dispatch.hpp`, [3663](#)
- probabilities
 - `std::discrete_distribution`, [2511](#)
- `probe_fn_base.hpp`, [3663](#)
- `profiler.h`, [3663](#)
- `profiler_algos.h`, [3665](#)
- `profiler_container_size.h`, [3666](#)
- `profiler_hash_func.h`, [3666](#)
- `profiler_hashtable_size.h`, [3667](#)
- `profiler_list_to_slist.h`, [3667](#)
- `profiler_list_to_vector.h`, [3668](#)
- `profiler_map_to_unordered_map.h`, [3669](#)
- `profiler_node.h`, [3669](#)
- `profiler_state.h`, [3670](#)
- `profiler_trace.h`, [3671](#)
- `profiler_vector_size.h`, [3673](#)
- `profiler_vector_to_list.h`, [3673](#)
- `propagate_const`, [3674](#)
- `propagate_on_container_copy_assignment`
 - `__gnu_cxx::__alloc_traits`, [715](#)
 - `std::allocator_traits`, [1650](#)
 - `std::allocator_traits< allocator< _Tp > >`, [1656](#)
- `propagate_on_container_move_assignment`
 - `__gnu_cxx::__alloc_traits`, [715](#)
 - `std::allocator_traits`, [1651](#)
 - `std::allocator_traits< allocator< _Tp > >`, [1656](#)
- `propagate_on_container_swap`
 - `__gnu_cxx::__alloc_traits`, [715](#)
 - `std::allocator_traits`, [1651](#)
 - `std::allocator_traits< allocator< _Tp > >`, [1656](#)
- ptr_fun
 - Adaptors for pointers to functions, [6](#)
- `ptr_traits.h`, [3676](#)
- pubimbue
 - `__gnu_cxx::enc_filebuf`, [823](#)
 - `__gnu_cxx::stdio_filebuf`, [882](#)
 - `__gnu_cxx::stdio_sync_filebuf`, [903](#)
 - `std::basic_filebuf`, [1711](#)
 - `std::basic_streambuf`, [2154](#)
 - `std::basic_stringbuf`, [2228](#)
 - `std::wbuffer_convert`, [3369](#)
- pubseekoff
 - `__gnu_cxx::enc_filebuf`, [824](#)
 - `__gnu_cxx::stdio_filebuf`, [882](#)
 - `__gnu_cxx::stdio_sync_filebuf`, [904](#)
 - `std::basic_filebuf`, [1711](#)
 - `std::basic_streambuf`, [2154](#)
 - `std::basic_stringbuf`, [2229](#)
 - `std::wbuffer_convert`, [3369](#)
- pubseekpos
 - `__gnu_cxx::enc_filebuf`, [824](#)
 - `__gnu_cxx::stdio_filebuf`, [883](#)
 - `__gnu_cxx::stdio_sync_filebuf`, [904](#)
 - `std::basic_filebuf`, [1712](#)
 - `std::basic_streambuf`, [2155](#)
 - `std::basic_stringbuf`, [2229](#)
 - `std::wbuffer_convert`, [3370](#)
- pubsetbuf
 - `__gnu_cxx::enc_filebuf`, [824](#)
 - `__gnu_cxx::stdio_filebuf`, [883](#)
 - `__gnu_cxx::stdio_sync_filebuf`, [904](#)
 - `std::basic_filebuf`, [1712](#)
 - `std::basic_streambuf`, [2155](#)
 - `std::basic_stringbuf`, [2229](#)
 - `std::wbuffer_convert`, [3370](#)
- pubsync
 - `__gnu_cxx::enc_filebuf`, [825](#)
 - `__gnu_cxx::stdio_filebuf`, [883](#)
 - `__gnu_cxx::stdio_sync_filebuf`, [904](#)
 - `std::basic_filebuf`, [1712](#)
 - `std::basic_streambuf`, [2155](#)

- std::basic_stringbuf, [2230](#)
- std::wbuffer_convert, [3370](#)
- push
 - std::priority_queue, [3039](#)
 - std::queue, [3045](#)
 - std::stack, [3133](#)
- push_back
 - __gnu_cxx::__versa_string, [779](#)
 - std::basic_string, [2207](#)
 - std::deque, [2501](#)
 - std::list, [2748](#)
 - std::tr2::dynamic_bitset, [3201](#)
 - std::vector, [3355](#)
- push_front
 - __gnu_parallel::__RestrictedBoundedConcurrentQueue, [1100](#)
 - std::deque, [2501](#)
 - std::forward_list, [2570](#)
 - std::list, [2748](#)
- push_heap
 - Heap, [93](#), [94](#)
- put
 - std::basic_fstream, [1764](#)
 - std::basic_istream, [1900](#)
 - std::basic_ofstream, [2042](#)
 - std::basic_ostream, [2079](#)
 - std::basic_ostringstream, [2119](#)
 - std::basic_stringstream, [2279](#)
 - std::money_put, [2850](#)
 - std::num_put, [2955](#)–[2960](#)
 - std::time_put, [3176](#), [3177](#)
 - std::time_put_byname, [3180](#)
- put_money
 - std, [632](#)
- put_time
 - std, [632](#)
- putback
 - std::basic_fstream, [1765](#)
 - std::basic_ifstream, [1815](#)
 - std::basic_istream, [1901](#)
 - std::basic_istream, [1949](#)
 - std::basic_istringstream, [1999](#)
 - std::basic_stringstream, [2280](#)
- pword
 - std::basic_fstream, [1765](#)
 - std::basic_ifstream, [1816](#)
 - std::basic_ios, [1847](#)
 - std::basic_istream, [1901](#)
 - std::basic_istream, [1950](#)
 - std::basic_istringstream, [2000](#)
 - std::basic_ofstream, [2043](#)
 - std::basic_ostream, [2080](#)
 - std::basic_ostringstream, [2120](#)
 - std::basic_stringstream, [2280](#)
 - std::ios_base, [2661](#)
- qsb_steals
 - __gnu_parallel::__Settings, [1107](#)
- quadratic_probe_fn_imp.hpp, [3677](#)
- queue, [3677](#)
 - std::queue, [3044](#)
- queue.h, [3677](#)
 - _GLIBCXX_VOLATILE, [3678](#)
- quicksort.h, [3678](#)
- quiet_NaN
 - std::numeric_limits, [2963](#)
- quoted
 - std, [632](#)
- quoted_string.h, [3678](#)
- r_erase_fn_imps.hpp, [3679](#)
- radix
 - std::__numeric_limits_base, [1571](#)
 - std::numeric_limits, [2967](#)
- random, [3679](#)
- Random Number Distributions, [253](#)
- Random Number Generation, [254](#)
 - generate_canonical, [254](#)
- Random Number Generators, [255](#)
 - minstd_rand, [256](#)
 - minstd_rand0, [256](#)
 - mt19937, [256](#)
 - mt19937_64, [257](#)
 - operator!=, [257](#)–[259](#)
 - operator<<, [259](#)
- Random Number Utilities, [261](#)
- random.h, [3680](#)
- random.tcc, [3684](#), [3688](#)
- random_number.h, [3690](#)
- random_sample
 - SGI, [307](#)
- random_sample_n
 - SGI, [307](#), [308](#)
- random_shuffle
 - Mutating, [157](#)
- random_shuffle.h, [3690](#)
- random_shuffle_minimal_n
 - __gnu_parallel::__Settings, [1107](#)
- range_access.h, [3691](#)
- ranged_hash_fn.hpp, [3693](#)
- ranged_probe_fn.hpp, [3693](#)
- ratio, [3694](#), [3695](#)
- ratio_divide
 - Rational Arithmetic, [263](#)
- ratio_multiply
 - Rational Arithmetic, [263](#)
- Rational Arithmetic, [262](#)
 - ratio_divide, [263](#)
 - ratio_multiply, [263](#)

- rb_tree, [3695](#)
- rb_tree_.hpp, [3696](#)
- rbegin
 - __gnu_cxx::__versa_string, [780](#)
 - std, [632](#), [633](#)
 - std::basic_string, [2207](#), [2208](#)
 - std::deque, [2501](#)
 - std::list, [2749](#)
 - std::map, [2805](#)
 - std::multimap, [2891](#)
 - std::multiset, [2914](#)
 - std::set, [3103](#)
 - std::vector, [3355](#)
- rc.hpp, [3696](#)
- rc_binomial_heap_.hpp, [3697](#)
- rc_string_base.h, [3697](#)
- rdbuf
 - std::basic_fstream, [1766](#)
 - std::basic_ifstream, [1816](#), [1817](#)
 - std::basic_ios, [1848](#)
 - std::basic_iostream, [1902](#)
 - std::basic_istream, [1950](#)
 - std::basic_istreamstream, [2000](#), [2001](#)
 - std::basic_ofstream, [2043](#), [2044](#)
 - std::basic_ostream, [2080](#), [2081](#)
 - std::basic_ostreamstream, [2120](#), [2121](#)
 - std::basic_stringstream, [2281](#)
- rdstate
 - std::basic_fstream, [1766](#)
 - std::basic_ifstream, [1817](#)
 - std::basic_ios, [1849](#)
 - std::basic_iostream, [1903](#)
 - std::basic_istream, [1951](#)
 - std::basic_istreamstream, [2001](#)
 - std::basic_ofstream, [2044](#)
 - std::basic_ostream, [2081](#)
 - std::basic_ostreamstream, [2121](#)
 - std::basic_stringstream, [2281](#)
- read
 - std::basic_fstream, [1767](#)
 - std::basic_ifstream, [1817](#)
 - std::basic_iostream, [1903](#)
 - std::basic_istream, [1951](#)
 - std::basic_istreamstream, [2001](#)
 - std::basic_stringstream, [2282](#)
- readsome
 - std::basic_fstream, [1767](#)
 - std::basic_ifstream, [1818](#)
 - std::basic_iostream, [1904](#)
 - std::basic_istream, [1952](#)
 - std::basic_istreamstream, [2002](#)
 - std::basic_stringstream, [2282](#)
- ready
 - std::match_results, [2818](#)
- rebind
 - std::pointer_traits, [3029](#)
- ref
 - std, [633](#), [634](#)
- reference
 - __gnu_pbds::detail::bin_search_tree_const_node_↔
it_, [1161](#)
 - __gnu_pbds::detail::bin_search_tree_node_it_, [1166](#)
 - __gnu_pbds::detail::binary_heap_const_iterator_,
[1174](#)
 - __gnu_pbds::detail::binary_heap_point_const_↔
iterator_, [1178](#)
 - __gnu_pbds::detail::left_child_next_sibling_heap_↔
const_iterator_, [1223](#)
 - __gnu_pbds::detail::left_child_next_sibling_heap_↔
node_point_const_iterator_, [1228](#)
 - const_iterator_, [1411](#)
 - iterator_, [1415](#)
 - point_const_iterator_, [1419](#)
 - point_iterator_, [1422](#)
 - std::back_insert_iterator, [1691](#)
 - std::front_insert_iterator, [2580](#)
 - std::insert_iterator, [2648](#)
 - std::istream_iterator, [2714](#)
 - std::istreambuf_iterator, [2717](#)
 - std::iterator, [2720](#)
 - std::ostream_iterator, [3001](#)
 - std::ostreambuf_iterator, [3004](#)
 - std::raw_storage_iterator, [3053](#)
 - std::set, [3085](#)
 - std::unordered_map, [3248](#)
 - std::unordered_multimap, [3272](#)
 - std::unordered_multiset, [3293](#)
 - std::unordered_set, [3316](#)
- regex, [3698](#)
 - Regular Expressions, [269](#)
- regex.h, [3698](#)
- regex.tcc, [3703](#)
- regex_automaton.h, [3704](#)
- regex_automaton.tcc, [3705](#)
- regex_compiler.h, [3705](#)
- regex_compiler.tcc, [3706](#)
- regex_constants.h, [3706](#)
- regex_error
 - std::regex_error, [3056](#)
- regex_error.h, [3708](#)
- regex_executor.h, [3709](#)
- regex_executor.tcc, [3709](#)
- regex_iterator
 - std::regex_iterator, [3057](#), [3058](#)
- regex_match
 - Regular Expressions, [287–290](#)
- regex_replace
 - Regular Expressions, [291–294](#)

- regex_scanner.h, [3709](#)
- regex_scanner.tcc, [3710](#)
- regex_search
 - Regular Expressions, [294–298](#)
- regex_token_iterator
 - std::regex_token_iterator, [3060–3062](#)
- regex_traits
 - std::regex_traits, [3065](#)
- register_callback
 - std::basic_fstream, [1768](#)
 - std::basic_ifstream, [1819](#)
 - std::basic_ios, [1849](#)
 - std::basic_iostream, [1905](#)
 - std::basic_istream, [1953](#)
 - std::basic_istreamstream, [2003](#)
 - std::basic_ofstream, [2044](#)
 - std::basic_ostream, [2082](#)
 - std::basic_ostreamstream, [2121](#)
 - std::basic_stringstream, [2283](#)
 - std::ios_base, [2662](#)
- Regular Expressions, [264](#)
 - cregex_token_iterator, [269](#)
 - csub_match, [269](#)
 - operator!=, [270–273](#)
 - operator<, [273–275](#)
 - operator<<, [275](#)
 - operator<=, [276–278](#)
 - operator>, [282–284](#)
 - operator>=, [285–287](#)
 - operator==, [278, 279, 281, 282](#)
 - regex, [269](#)
 - regex_match, [287–290](#)
 - regex_replace, [291–294](#)
 - regex_search, [294–298](#)
 - sregex_token_iterator, [269](#)
 - ssub_match, [269](#)
 - swap, [298, 299](#)
 - wcregex_token_iterator, [269](#)
 - wcsub_match, [270](#)
 - wregex, [270](#)
 - wsregex_token_iterator, [270](#)
 - wssub_match, [270](#)
- rehash
 - std::unordered_map, [3266](#)
 - std::unordered_multimap, [3287](#)
 - std::unordered_multiset, [3309](#)
 - std::unordered_set, [3332](#)
- release
 - std::auto_ptr, [1688](#)
 - std::unique_ptr, [3237](#)
 - std::unique_ptr< _Tp[], _Dp >, [3242](#)
- remove
 - Mutating, [157](#)
 - std::forward_list, [2571](#)
- std::list, [2749](#)
- remove_copy
 - Mutating, [157](#)
- remove_copy_if
 - Mutating, [158](#)
- remove_if
 - Mutating, [158](#)
 - std::forward_list, [2571](#)
 - std::list, [2749](#)
- rend
 - __gnu_cxx::__versa_string, [780](#)
 - std, [634, 635](#)
 - std::basic_string, [2208](#)
 - std::deque, [2502](#)
 - std::list, [2750](#)
 - std::map, [2805](#)
 - std::multimap, [2891](#)
 - std::multiset, [2914](#)
 - std::set, [3103](#)
 - std::vector, [3356](#)
- replace
 - __gnu_cxx::__versa_string, [780–785, 787, 788](#)
 - __gnu_debug::basic_string, [1003–1007](#)
 - Mutating, [159](#)
 - std::basic_string, [2208–2214](#)
- replace_copy
 - std, [635](#)
- replace_copy_if
 - Mutating, [159](#)
- replace_if
 - Mutating, [160](#)
- replace_minimal_n
 - __gnu_parallel::Settings, [1107](#)
- requested_size
 - __gnu_cxx::temporary_buffer, [918](#)
 - std::_Temporary_buffer, [1634](#)
- reserve
 - __gnu_cxx::__versa_string, [788](#)
 - __gnu_debug::basic_string, [1007](#)
 - std::basic_string, [2215](#)
 - std::unordered_map, [3266](#)
 - std::unordered_multimap, [3288](#)
 - std::unordered_multiset, [3310](#)
 - std::unordered_set, [3333](#)
 - std::vector, [3356](#)
- reset
 - std::auto_ptr, [1688](#)
 - std::bernoulli_distribution, [2302](#)
 - std::binomial_distribution, [2312](#)
 - std::bitset, [2323, 2324](#)
 - std::cauchy_distribution, [2329](#)
 - std::chi_squared_distribution, [2337](#)
 - std::discrete_distribution, [2512](#)
 - std::exponential_distribution, [2546](#)

- std::extreme_value_distribution, 2549
- std::fisher_f_distribution, 2553
- std::gamma_distribution, 2600
- std::geometric_distribution, 2604
- std::lognormal_distribution, 2779
- std::negative_binomial_distribution, 2921
- std::normal_distribution, 2927
- std::piecewise_constant_distribution, 3017
- std::piecewise_linear_distribution, 3022
- std::poisson_distribution, 3034
- std::student_t_distribution, 3137
- std::tr2::dynamic_bitset, 3201
- std::uniform_int_distribution, 3227
- std::uniform_real_distribution, 3230
- std::unique_ptr, 3237
- std::unique_ptr< _Tp[], _Dp >, 3242
- std::weibull_distribution, 3383
- resetiosflags
 - std, 635
- resize
 - __gnu_cxx::__versa_string, 789
 - __gnu_pbds::hash_standard_resize_policy, 1333
 - Numeric Arrays, 223
 - std::basic_string, 2215, 2216
 - std::deque, 2502
 - std::forward_list, 2571, 2572
 - std::list, 2750
 - std::tr2::dynamic_bitset, 3203
 - std::vector, 3356, 3357
- resize_fn_imps.hpp, 3710
- resize_no_store_hash_fn_imps.hpp, 3710, 3711
- resize_policy.hpp, 3711
- resize_store_hash_fn_imps.hpp, 3711
- result_type
 - __gnu_cxx::__detail::__Ffit_finder, 721
 - __gnu_cxx::binary_compose, 805
 - __gnu_cxx::project1st, 852
 - __gnu_cxx::project2nd, 853
 - __gnu_cxx::select1st, 866
 - __gnu_cxx::select2nd, 867
 - __gnu_cxx::subtractive_rng, 915
 - __gnu_cxx::unary_compose, 928
 - __gnu_parallel::__EqualFromLess, 1053
 - __gnu_parallel::__EqualTo, 1054
 - __gnu_parallel::__Less, 1062
 - __gnu_parallel::__Lexicographic, 1064
 - __gnu_parallel::__LexicographicReverse, 1065
 - __gnu_parallel::__Multiplies, 1088
 - __gnu_parallel::__Plus, 1091
 - __gnu_parallel::__binder1st, 1015
 - __gnu_parallel::__binder2nd, 1016
 - __gnu_parallel::__unary_negate, 1048
 - std::_Maybe_unary_or_binary_function< _Res, _T1, _T2 >, 1625
 - std::_Maybe_unary_or_binary_function< _Res, _T1, _T2 >, 1626
 - std::bernoulli_distribution, 2300
 - std::binary_function, 2304
 - std::binary_negate, 2306
 - std::binder1st, 2308
 - std::binder2nd, 2309
 - std::binomial_distribution, 2311
 - std::cauchy_distribution, 2328
 - std::chi_squared_distribution, 2336
 - std::const_mem_fun1_ref_t, 2393
 - std::const_mem_fun1_t, 2394
 - std::const_mem_fun_ref_t, 2396
 - std::const_mem_fun_t, 2397
 - std::discard_block_engine, 2505
 - std::discrete_distribution, 2510
 - std::divides, 2514
 - std::equal_to, 2518
 - std::experimental::fundamentals_v2::owner_less< shared_ptr< _Tp > >, 2540
 - std::experimental::fundamentals_v2::owner_less< weak_ptr< _Tp > >, 2541
 - std::exponential_distribution, 2544
 - std::extreme_value_distribution, 2548
 - std::fisher_f_distribution, 2552
 - std::gamma_distribution, 2598
 - std::geometric_distribution, 2603
 - std::greater, 2606
 - std::greater_equal, 2607
 - std::hash< __gnu_cxx::throw_value_limit >, 2616
 - std::hash< __gnu_cxx::throw_value_random >, 2617
 - std::independent_bits_engine, 2639
 - std::less, 2723
 - std::less_equal, 2725
 - std::linear_congruential_engine, 2728
 - std::logical_and, 2772
 - std::logical_not, 2774
 - std::logical_or, 2776
 - std::lognormal_distribution, 2778
 - std::mem_fun1_ref_t, 2820
 - std::mem_fun1_t, 2822
 - std::mem_fun_ref_t, 2823
 - std::mem_fun_t, 2824
 - std::mersenne_twister_engine, 2827
 - std::minus, 2837
 - std::modulus, 2839
 - std::multiplies, 2895
 - std::negate, 2918
 - std::negative_binomial_distribution, 2920
 - std::normal_distribution, 2925
 - std::not_equal_to, 2929
 - std::owner_less< shared_ptr< _Tp > >, 3009
 - std::owner_less< weak_ptr< _Tp > >, 3010

- std::piecewise_constant_distribution, [3016](#)
- std::piecewise_linear_distribution, [3021](#)
- std::plus, [3025](#)
- std::pointer_to_binary_function, [3027](#)
- std::pointer_to_unary_function, [3028](#)
- std::poisson_distribution, [3033](#)
- std::random_device, [3047](#)
- std::seed_seq, [3080](#)
- std::shuffle_order_engine, [3125](#)
- std::student_t_distribution, [3136](#)
- std::subtract_with_carry_engine, [3144](#)
- std::unary_function, [3222](#)
- std::unary_negate, [3223](#)
- std::uniform_int_distribution, [3226](#)
- std::uniform_real_distribution, [3229](#)
- std::weibull_distribution, [3381](#)
- rethrow_exception
 - Exceptions, [75](#)
- rethrow_if_nested
 - Exceptions, [75](#)
- return_temporary_buffer
 - std, [636](#)
- reverse
 - Mutating, [160](#)
 - std::forward_list, [2572](#)
 - std::list, [2751](#)
- reverse_copy
 - Mutating, [161](#)
- reverse_iteration
 - __gnu_pbds::container_traits, [1151](#)
- reverse_iterator
 - std::reverse_iterator, [3073](#)
 - std::set, [3085](#)
- rfind
 - __gnu_cxx::__versa_string, [790](#), [791](#)
 - __gnu_debug::basic_string, [1008](#)
 - std::basic_string, [2216](#), [2217](#)
- riemann_zeta
 - Mathematical Special Functions, [134](#), [143](#)
- riemann_zetaf
 - Mathematical Special Functions, [135](#)
- riemann_zetal
 - Mathematical Special Functions, [135](#)
- right
 - std, [636](#)
 - std::basic_fstream, [1782](#)
 - std::basic_ifstream, [1831](#)
 - std::basic_ios, [1858](#)
 - std::basic_iostream, [1918](#)
 - std::basic_istream, [1965](#)
 - std::basic_istreamstream, [2015](#)
 - std::basic_ofstream, [2055](#)
 - std::basic_ostream, [2093](#)
 - std::basic_ostreamstream, [2133](#)
 - std::basic_stringstream, [2298](#)
 - std::ios_base, [2669](#)
- rope, [3711](#)
- ropeimpl.h, [3715](#)
- rotate
 - Mutating, [161](#)
- rotate_copy
 - Mutating, [162](#)
- rotate_fn_imps.hpp, [3715](#)
- round_error
 - std::numeric_limits, [2964](#)
- round_style
 - std::__numeric_limits_base, [1571](#)
 - std::numeric_limits, [2967](#)
- round_to_nearest
 - std, [562](#)
- round_toward_infinity
 - std, [562](#)
- round_toward_neg_infinity
 - std, [562](#)
- round_toward_zero
 - std, [562](#)
- runtime_error
 - std::runtime_error, [3077](#)
- SGL, [300](#)
 - _Find_first, [303](#)
 - _Find_next, [303](#)
 - _Unchecked_flip, [304](#)
 - _Unchecked_reset, [304](#)
 - _Unchecked_set, [304](#)
 - _Unchecked_test, [304](#)
 - __median, [302](#), [303](#)
 - compose1, [305](#)
 - compose2, [305](#)
 - constant0, [305](#)
 - constant1, [305](#)
 - constant2, [305](#)
 - copy_n, [305](#)
 - distance, [306](#)
 - identity_element, [306](#)
 - lexicographical_compare_3way, [306](#)
 - power, [307](#)
 - random_sample, [307](#)
 - random_sample_n, [307](#), [308](#)
 - uninitialized_copy_n, [308](#)
- safe_base.h, [3716](#)
- safe_container.h, [3716](#)
- safe_iterator.h, [3716](#)
- safe_iterator.tcc, [3718](#)
- safe_local_iterator.h, [3719](#)
- safe_local_iterator.tcc, [3719](#)
- safe_sequence.h, [3720](#)
- safe_sequence.tcc, [3720](#)

- safe_unordered_base.h, [3721](#)
- safe_unordered_container.h, [3721](#)
- safe_unordered_container.tcc, [3721](#)
- sample
 - experimental/algorithm, [3403](#)
- sample_probe_fn
 - __gnu_pbds::sample_probe_fn, [1353](#)
- sample_probe_fn.hpp, [3722](#)
- sample_range_hashing
 - __gnu_pbds::sample_range_hashing, [1354](#)
 - __gnu_pbds::sample_resize_policy, [1359](#)
 - __gnu_pbds::sample_resize_trigger, [1362](#)
 - __gnu_pbds::sample_size_policy, [1363](#)
- sample_range_hashing.hpp, [3722](#)
- sample_ranged_hash_fn
 - __gnu_pbds::sample_ranged_hash_fn, [1356](#)
- sample_ranged_hash_fn.hpp, [3722](#)
- sample_ranged_probe_fn.hpp, [3723](#)
- sample_resize_policy
 - __gnu_pbds::sample_resize_policy, [1358](#)
- sample_resize_policy.hpp, [3723](#)
- sample_resize_trigger
 - __gnu_pbds::sample_resize_trigger, [1361](#)
- sample_resize_trigger.hpp, [3723](#)
- sample_size_policy
 - __gnu_pbds::sample_size_policy, [1363](#)
- sample_size_policy.hpp, [3724](#)
- sample_tree_node_update.hpp, [3724](#)
- sample_trie_access_traits.hpp, [3724](#)
- sample_trie_node_update
 - __gnu_pbds::sample_trie_node_update, [1366](#)
- sample_trie_node_update.hpp, [3725](#)
- sample_update_policy
 - __gnu_pbds::sample_update_policy, [1367](#)
- sample_update_policy.hpp, [3725](#)
- sbumpc
 - __gnu_cxx::enc_filebuf, [825](#)
 - __gnu_cxx::stdio_filebuf, [883](#)
 - __gnu_cxx::stdio_sync_filebuf, [904](#)
 - std::basic_filebuf, [1712](#)
 - std::basic_streambuf, [2155](#)
 - std::basic_stringbuf, [2230](#)
 - std::wbuffer_convert, [3370](#)
- scan_is
 - std::__ctype_abstract_base, [1444](#)
 - std::ctype, [2408](#)
 - std::ctype< char >, [2421](#)
 - std::ctype< wchar_t >, [2437](#)
 - std::ctype_byname, [2452](#)
 - std::ctype_byname< char >, [2465](#)
- scan_not
 - std::__ctype_abstract_base, [1445](#)
 - std::ctype, [2408](#)
 - std::ctype< char >, [2422](#)
- std::ctype< wchar_t >, [2438](#)
- std::ctype_byname, [2453](#)
- std::ctype_byname< char >, [2466](#)
- scientific
 - std, [636](#)
 - std::basic_fstream, [1782](#)
 - std::basic_ifstream, [1831](#)
 - std::basic_ios, [1858](#)
 - std::basic_iostream, [1918](#)
 - std::basic_istream, [1965](#)
 - std::basic_istreamstring, [2015](#)
 - std::basic_ofstream, [2055](#)
 - std::basic_ostream, [2093](#)
 - std::basic_ostreamstring, [2133](#)
 - std::basic_stringstream, [2298](#)
 - std::ios_base, [2669](#)
- scoped_allocator, [3725](#)
- search
 - Non-Mutating, [186](#)
- search.h, [3726](#)
- search_minimal_n
 - __gnu_parallel::Settings, [1108](#)
- search_n
 - Non-Mutating, [187](#)
- second
 - __gnu_parallel::IteratorPair, [1059](#)
 - std::pair, [3014](#)
 - std::sub_match, [3142](#)
- second_argument_type
 - __gnu_cxx::project1st, [852](#)
 - __gnu_cxx::project2nd, [853](#)
 - __gnu_parallel::EqualFromLess, [1053](#)
 - __gnu_parallel::EqualTo, [1054](#)
 - __gnu_parallel::Less, [1063](#)
 - __gnu_parallel::Lexicographic, [1064](#)
 - __gnu_parallel::LexicographicReverse, [1065](#)
 - __gnu_parallel::Multiplies, [1088](#)
 - __gnu_parallel::Plus, [1091](#)
 - std::_Maybe_unary_or_binary_function< _Res, _T1, _T2 >, [1626](#)
 - std::binary_function, [2305](#)
 - std::binary_negate, [2306](#)
 - std::const_mem_fun1_ref_t, [2393](#)
 - std::const_mem_fun1_t, [2395](#)
 - std::divides, [2515](#)
 - std::equal_to, [2518](#)
 - std::experimental::fundamentals_v2::owner_less< shared_ptr< _Tp > >, [2540](#)
 - std::experimental::fundamentals_v2::owner_less< weak_ptr< _Tp > >, [2541](#)
 - std::greater, [2606](#)
 - std::greater_equal, [2608](#)
 - std::less, [2724](#)
 - std::less_equal, [2726](#)

- std::logical_and, 2772
- std::logical_or, 2776
- std::mem_fun1_ref_t, 2821
- std::mem_fun1_t, 2822
- std::minus, 2837
- std::modulus, 2839
- std::multiplies, 2895
- std::not_equal_to, 2930
- std::owner_less< shared_ptr< _Tp > >, 3009
- std::owner_less< weak_ptr< _Tp > >, 3010
- std::plus, 3025
- std::pointer_to_binary_function, 3027
- second_type
 - __gnu_parallel::_iteratorPair, 1059
 - std::pair, 3013
 - std::sub_match, 3140
- seconds
 - std::chrono, 682
- seed
 - std::discard_block_engine, 2507
 - std::independent_bits_engine, 2641, 2642
 - std::linear_congruential_engine, 2729
 - std::shuffle_order_engine, 3127
 - std::subtract_with_carry_engine, 3145, 3146
- seed_seq
 - std::seed_seq, 3081
- seekdir
 - std::basic_fstream, 1733
 - std::basic_ifstream, 1792
 - std::basic_ios, 1839
 - std::basic_iostream, 1868
 - std::basic_istream, 1927
 - std::basic_istreamstream, 1976
 - std::basic_ofstream, 2024
 - std::basic_ostream, 2064
 - std::basic_ostreamstream, 2103
 - std::basic_stringstream, 2248
 - std::ios_base, 2658
- seekg
 - std::basic_fstream, 1768, 1769
 - std::basic_ifstream, 1819, 1820
 - std::basic_iostream, 1905
 - std::basic_istream, 1953, 1954
 - std::basic_istreamstream, 2003, 2004
 - std::basic_stringstream, 2283, 2284
- seekoff
 - __gnu_cxx::enc_filebuf, 825
 - __gnu_cxx::stdio_filebuf, 884
 - __gnu_cxx::stdio_sync_filebuf, 905
 - std::basic_filebuf, 1713
 - std::basic_streambuf, 2156
 - std::basic_stringbuf, 2230
 - std::wbuffer_convert, 3370
- seekp
 - std::basic_fstream, 1770
 - std::basic_iostream, 1906
 - std::basic_ofstream, 2045
 - std::basic_ostream, 2082
 - std::basic_ostreamstream, 2122
 - std::basic_stringstream, 2285
- seekpos
 - __gnu_cxx::enc_filebuf, 825
 - __gnu_cxx::stdio_filebuf, 884
 - __gnu_cxx::stdio_sync_filebuf, 905
 - std::basic_filebuf, 1713
 - std::basic_streambuf, 2156
 - std::basic_stringbuf, 2230
 - std::wbuffer_convert, 3371
- select_on_container_copy_construction
 - __gnu_cxx::_alloc_traits, 718
 - std::allocator_traits, 1654
 - std::allocator_traits< allocator< _Tp > >, 1658
- sentry
 - std::basic_istream::sentry, 1967
 - std::basic_ostream::sentry, 2095
- Sequences, 309
- sequential
 - __gnu_parallel, 410
- set, 3726, 3727
 - __gnu_parallel::_Settings, 1103
 - std::bitset, 2324
 - std::set, 3086–3089
 - std::tr2::dynamic_bitset, 3203
- Set Operation, 310
 - includes, 311
 - set_difference, 311, 312
 - set_intersection, 312, 313
 - set_symmetric_difference, 313, 314
 - set_union, 314, 315
- set.h, 3728, 3729
- set_difference
 - Set Operation, 311, 312
- set_difference_minimal_n
 - __gnu_parallel::_Settings, 1108
- set_intersection
 - Set Operation, 312, 313
- set_intersection_minimal_n
 - __gnu_parallel::_Settings, 1108
- set_load
 - __gnu_pbds::cc_hash_max_collision_check ↔
resize_trigger, 1143
- set_loads
 - __gnu_pbds::hash_load_check_resize_trigger, 1329
- set_new_handler
 - std, 636
- set_num_threads
 - __gnu_parallel::balanced_quicksort_tag, 1111
 - __gnu_parallel::balanced_tag, 1112

- __gnu_parallel::default_parallel_tag, 1114
- __gnu_parallel::exact_tag, 1116
- __gnu_parallel::multiway_mergesort_exact_tag, 1119
- __gnu_parallel::multiway_mergesort_sampling_tag, 1120
- __gnu_parallel::multiway_mergesort_tag, 1122
- __gnu_parallel::omp_loop_static_tag, 1123
- __gnu_parallel::omp_loop_tag, 1124
- __gnu_parallel::parallel_tag, 1127
- __gnu_parallel::quicksort_tag, 1129
- __gnu_parallel::sampling_tag, 1130
- __gnu_parallel::unbalanced_tag, 1131
- set_operations.h, 3729
- set_symmetric_difference
 - Set Operation, 313, 314
- set_symmetric_difference_minimal_n
 - __gnu_parallel::_Settings, 1108
- set_terminate
 - Exceptions, 75
- set_unexpected
 - Exceptions, 75
- set_union
 - Set Operation, 314, 315
- set_union_minimal_n
 - __gnu_parallel::_Settings, 1108
- setbase
 - std, 636
- setbuf
 - __gnu_cxx::enc_filebuf, 826
 - __gnu_cxx::stdio_filebuf, 884
 - __gnu_cxx::stdio_sync_filebuf, 905
 - std::basic_filebuf, 1713
 - std::basic_streambuf, 2156
 - std::basic_stringbuf, 2231
 - std::wbuffer_convert, 3371
- setf
 - std::basic_fstream, 1771
 - std::basic_ifstream, 1820, 1821
 - std::basic_ios, 1849, 1850
 - std::basic_iostream, 1907
 - std::basic_istream, 1954, 1955
 - std::basic_istreamstream, 2004, 2005
 - std::basic_ofstream, 2045, 2046
 - std::basic_ostream, 2083
 - std::basic_ostreamstream, 2123
 - std::basic_stringstream, 2286
 - std::ios_base, 2662, 2663
- setfill
 - std, 637
- setg
 - __gnu_cxx::enc_filebuf, 826
 - __gnu_cxx::stdio_filebuf, 885
 - __gnu_cxx::stdio_sync_filebuf, 906
 - std::basic_filebuf, 1714
 - std::basic_streambuf, 2157
 - std::basic_stringbuf, 2231
 - std::wbuffer_convert, 3371
- setiosflags
 - std, 637
- setp
 - __gnu_cxx::enc_filebuf, 827
 - __gnu_cxx::stdio_filebuf, 885
 - __gnu_cxx::stdio_sync_filebuf, 906
 - std::basic_filebuf, 1714
 - std::basic_streambuf, 2157
 - std::basic_stringbuf, 2232
 - std::wbuffer_convert, 3372
- setprecision
 - std, 637
- setstate
 - std::basic_fstream, 1771
 - std::basic_ifstream, 1821
 - std::basic_ios, 1850
 - std::basic_iostream, 1908
 - std::basic_istream, 1955
 - std::basic_istreamstream, 2005
 - std::basic_ofstream, 2046
 - std::basic_ostream, 2084
 - std::basic_ostreamstream, 2123
 - std::basic_stringstream, 2286
- settings.h, 3730
 - _GLIBCXX_PARALLEL_CONDITION, 3731
- setw
 - std, 637
- sgetc
 - __gnu_cxx::enc_filebuf, 827
 - __gnu_cxx::stdio_filebuf, 886
 - __gnu_cxx::stdio_sync_filebuf, 907
 - std::basic_filebuf, 1715
 - std::basic_streambuf, 2158
 - std::basic_stringbuf, 2232
 - std::wbuffer_convert, 3372
- sgetn
 - __gnu_cxx::enc_filebuf, 827
 - __gnu_cxx::stdio_filebuf, 886
 - __gnu_cxx::stdio_sync_filebuf, 907
 - std::basic_filebuf, 1715
 - std::basic_streambuf, 2158
 - std::basic_stringbuf, 2232
 - std::wbuffer_convert, 3372
- shared_future
 - std::shared_future, 3108
 - std::shared_future< _Res & >, 3110, 3111
 - std::shared_future< void >, 3113
- shared_mutex, 3732
- shared_ptr
 - std::shared_ptr, 3116–3121

- shared_ptr.h, [3732](#), [3734](#)
- shared_ptr_atomic.h, [3737](#)
- shared_ptr_base.h, [3738](#)
- shift
 - Numeric Arrays, [223](#)
- showbase
 - std, [638](#)
 - std::basic_fstream, [1783](#)
 - std::basic_ifstream, [1831](#)
 - std::basic_ios, [1859](#)
 - std::basic_istream, [1919](#)
 - std::basic_istream, [1965](#)
 - std::basic_istreamstream, [2015](#)
 - std::basic_ofstream, [2056](#)
 - std::basic_ostream, [2093](#)
 - std::basic_ostreamstream, [2133](#)
 - std::basic_stringstream, [2298](#)
 - std::ios_base, [2670](#)
- showmany
 - __gnu_cxx::enc_filebuf, [828](#)
 - __gnu_cxx::stdio_filebuf, [887](#)
 - __gnu_cxx::stdio_sync_filebuf, [907](#)
 - std::basic_filebuf, [1716](#)
 - std::basic_streambuf, [2158](#)
 - std::basic_stringbuf, [2233](#)
 - std::wbuffer_convert, [3373](#)
- showpoint
 - std, [638](#)
 - std::basic_fstream, [1783](#)
 - std::basic_ifstream, [1831](#)
 - std::basic_ios, [1859](#)
 - std::basic_istream, [1919](#)
 - std::basic_istream, [1965](#)
 - std::basic_istreamstream, [2015](#)
 - std::basic_ofstream, [2056](#)
 - std::basic_ostream, [2093](#)
 - std::basic_ostreamstream, [2133](#)
 - std::basic_stringstream, [2298](#)
 - std::ios_base, [2670](#)
- showpos
 - std, [638](#)
 - std::basic_fstream, [1783](#)
 - std::basic_ifstream, [1831](#)
 - std::basic_ios, [1859](#)
 - std::basic_istream, [1919](#)
 - std::basic_istream, [1965](#)
 - std::basic_istreamstream, [2016](#)
 - std::basic_ofstream, [2056](#)
 - std::basic_ostream, [2093](#)
 - std::basic_ostreamstream, [2134](#)
 - std::basic_stringstream, [2298](#)
 - std::ios_base, [2670](#)
- shrink_to_fit
 - __gnu_cxx::__versa_string, [792](#)
 - std::basic_string, [2218](#)
 - std::deque, [2503](#)
 - std::vector, [3357](#)
- shuffle
 - Mutating, [162](#)
- shuffle_order_engine
 - std::shuffle_order_engine, [3125](#), [3126](#)
- signaling_NaN
 - std::numeric_limits, [2964](#)
- sin
 - Complex Numbers, [56](#)
- sinh
 - Complex Numbers, [56](#)
- size
 - __gnu_cxx::__versa_string, [792](#)
 - __gnu_cxx::temporary_buffer, [918](#)
 - __gnu_debug::basic_string, [1008](#)
 - Numeric Arrays, [224](#)
 - std::_Temporary_buffer, [1634](#)
 - std::basic_string, [2218](#)
 - std::bitset, [2324](#)
 - std::deque, [2503](#)
 - std::list, [2751](#)
 - std::map, [2805](#)
 - std::match_results, [2818](#)
 - std::multimap, [2892](#)
 - std::multiset, [2914](#)
 - std::priority_queue, [3040](#)
 - std::queue, [3046](#)
 - std::set, [3103](#)
 - std::stack, [3133](#)
 - std::tr2::dynamic_bitset, [3203](#)
 - std::unordered_map, [3266](#)
 - std::unordered_multimap, [3288](#)
 - std::unordered_multiset, [3310](#)
 - std::unordered_set, [3333](#)
 - std::vector, [3357](#)
- size_fn_imps.hpp, [3740](#)
- size_type
 - __gnu_pbds::hash_prime_size_policy, [1330](#)
 - __gnu_pbds::sample_range_hashing, [1354](#)
 - __gnu_pbds::sample_resize_policy, [1358](#)
 - __gnu_pbds::sample_resize_trigger, [1360](#)
 - __gnu_pbds::sample_size_policy, [1363](#)
 - __gnu_pbds::trie_prefix_search_node_update, [1384](#)
 - std::allocator_traits, [1651](#)
 - std::allocator_traits< allocator< _Tp > >, [1656](#)
 - std::set, [3086](#)
 - std::unordered_map, [3248](#)
 - std::unordered_multimap, [3272](#)
 - std::unordered_multiset, [3294](#)
 - std::unordered_set, [3316](#)
- skipws
 - std, [638](#)

- std::basic_fstream, 1783
- std::basic_ifstream, 1831
- std::basic_ios, 1859
- std::basic_iostream, 1919
- std::basic_istream, 1965
- std::basic_istreamstream, 2016
- std::basic_ofstream, 2056
- std::basic_ostream, 2093
- std::basic_ostreamstream, 2134
- std::basic_stringstream, 2298
- std::ios_base, 2670
- sleep_for
 - std::this_thread, 707
- sleep_until
 - std::this_thread, 707
- slice
 - Numeric Arrays, 201
- slice_array
 - Numeric Arrays, 202
- slice_array.h, 3740
- slist, 3741
- snextc
 - __gnu_cxx::enc_filebuf, 828
 - __gnu_cxx::stdio_filebuf, 887
 - __gnu_cxx::stdio_sync_filebuf, 908
 - std::basic_filebuf, 1716
 - std::basic_streambuf, 2159
 - std::basic_stringbuf, 2233
 - std::wbuffer_convert, 3373
- sort
 - Sorting, 332, 333
 - std::forward_list, 2572
 - std::list, 2751
- sort.h, 3742
- sort_heap
 - Heap, 94
- sort_minimal_n
 - __gnu_parallel::Settings, 1108
- sort_mwms_oversampling
 - __gnu_parallel::Settings, 1108
- sort_qs_num_samples_preset
 - __gnu_parallel::Settings, 1108
- sort_qsb_base_case_maximal_n
 - __gnu_parallel::Settings, 1109
- Sorting, 316
 - inplace_merge, 318
 - is_sorted, 319
 - is_sorted_until, 320
 - lexicographical_compare, 320, 321
 - max, 321, 322
 - max_element, 322, 323
 - merge, 323, 324
 - min, 324, 325
 - min_element, 325, 326
 - minmax, 326
 - minmax_element, 327
 - next_permutation, 328
 - nth_element, 329
 - partial_sort, 330
 - partial_sort_copy, 331
 - prev_permutation, 332
 - sort, 332, 333
 - stable_sort, 333, 334
- specfun.h, 3742
- sph_bessel
 - Mathematical Special Functions, 135, 144
- sph_besself
 - Mathematical Special Functions, 136
- sph_bessell
 - Mathematical Special Functions, 136
- sph_legendre
 - Mathematical Special Functions, 136, 144
- sph_legendref
 - Mathematical Special Functions, 137
- sph_legendrel
 - Mathematical Special Functions, 137
- sph_neumann
 - Mathematical Special Functions, 137, 144
- sph_neumannf
 - Mathematical Special Functions, 138
- sph_neumannl
 - Mathematical Special Functions, 138
- splay_fn_imps.hpp, 3745
- splay_tree.hpp, 3745
- splice
 - std::list, 2751–2753
- splice_after
 - std::forward_list, 2573, 2575
- split_fn_imps.hpp, 3745
- split_join_can_throw
 - __gnu_pbds::container_traits, 1151
- split_join_fn_imps.hpp, 3746, 3747
- sputbackc
 - __gnu_cxx::enc_filebuf, 828
 - __gnu_cxx::stdio_filebuf, 887
 - __gnu_cxx::stdio_sync_filebuf, 908
 - std::basic_filebuf, 1716
 - std::basic_streambuf, 2159
 - std::basic_stringbuf, 2233
 - std::wbuffer_convert, 3373
- sputc
 - __gnu_cxx::enc_filebuf, 829
 - __gnu_cxx::stdio_filebuf, 888
 - __gnu_cxx::stdio_sync_filebuf, 909
 - std::basic_filebuf, 1717
 - std::basic_streambuf, 2160
 - std::basic_stringbuf, 2234
 - std::wbuffer_convert, 3374

sputn
 __gnu_cxx::enc_filebuf, 829
 __gnu_cxx::stdio_filebuf, 888
 __gnu_cxx::stdio_sync_filebuf, 909
 std::basic_filebuf, 1717
 std::basic_streambuf, 2160
 std::basic_stringbuf, 2234
 std::wbuffer_convert, 3374
 sqrt
 Complex Numbers, 56
 sregex_token_iterator
 Regular Expressions, 269
 sso_string_base.h, 3747
 sstream, 3747
 sstream.tcc, 3748
 ssub_match
 Regular Expressions, 269
 stable_partition
 Mutating, 163
 stable_sort
 Sorting, 333, 334
 stack, 3748
 std::stack, 3133
 standard_policies.hpp, 3749
 start
 Numeric Arrays, 224
 state
 std::fpos, 2578
 std::wbuffer_convert, 3375
 std::wstring_convert, 3387
 static_pointer_cast
 std, 638
 std, 458
 _Construct, 570
 _Destroy, 570
 __allocate_guarded, 562
 __final_insertion_sort, 562
 __find_if, 562
 __find_if_not, 563
 __find_if_not_n, 563
 __gcd, 563
 __heap_select, 563
 __inplace_stable_sort, 563
 __insertion_sort, 564
 __introsort_loop, 564
 __invoke, 564
 __ioinit, 645
 __lg, 564
 __merge_adaptive, 564
 __merge_without_buffer, 565
 __move_median_to_first, 565
 __move_merge, 565
 __move_merge_adaptive, 565
 __move_merge_adaptive_backward, 565
 __once_call, 645
 __once_callable, 646
 __once_proxy, 566
 __partition, 566
 __ptr_rebind, 559
 __reverse, 567
 __rotate_adaptive, 567
 __search_n_aux, 567
 __stable_partition_adaptive, 568
 __try_to_lock, 568
 __umap_traits, 559
 __ummap_traits, 559
 __umset_traits, 559
 __unguarded_insertion_sort, 568
 __unguarded_linear_insert, 569
 __unguarded_partition, 569
 __unguarded_partition_pivot, 569
 __unique_copy, 569, 570
 __uset_traits, 559
 accumulate, 571
 acos, 572
 acosh, 572
 adjacent_difference, 572
 advance, 573
 align, 573
 arg, 574
 asin, 574
 asinh, 574
 atan, 574
 atanh, 574
 begin, 574, 575
 boolalpha, 576
 call_once, 576
 cbegin, 576
 cend, 576
 cerr, 646
 cin, 646
 clog, 646
 const_pointer_cast, 577
 cout, 646
 crbegin, 577
 cref, 577
 crend, 577
 dec, 578
 defaultfloat, 578
 denorm_absent, 561
 denorm_indeterminate, 561
 denorm_present, 561
 distance, 578
 dynamic_pointer_cast, 579
 end, 579, 580
 endl, 580
 ends, 580
 exchange, 581

fabs, [581](#)
fixed, [581](#)
float_denorm_style, [561](#)
float_round_style, [561](#)
flush, [581](#)
get_money, [581](#)
get_new_handler, [582](#)
get_temporary_buffer, [582](#)
get_time, [582](#)
getline, [583–585](#)
hex, [585](#)
hexfloat, [585](#)
index_sequence, [560](#)
index_sequence_for, [560](#)
inner_product, [585](#), [586](#)
internal, [586](#)
io_errc, [562](#)
iota, [586](#)
isalnum, [588](#)
isalpha, [588](#)
isblank, [588](#)
isctrl, [588](#)
isdigit, [588](#)
isgraph, [588](#)
islower, [589](#)
isprint, [589](#)
ispunct, [589](#)
isspace, [589](#)
isupper, [589](#)
isxdigit, [589](#)
left, [590](#)
lock, [590](#)
make_index_sequence, [560](#)
make_integer_sequence, [560](#)
new_handler, [560](#)
noboolalpha, [590](#)
noshowbase, [591](#)
noshowpoint, [591](#)
noshowpos, [591](#)
noskipws, [591](#)
nounitbuf, [591](#)
nouppercase, [591](#)
oct, [592](#)
operator!=, [592–594](#)
operator<, [597–603](#)
operator<<, [603–609](#)
operator<=, [609–612](#)
operator>, [618–621](#)
operator>>, [623–629](#)
operator>=, [621–623](#)
operator^, [630](#)
operator+, [595–597](#)
operator==, [612–618](#)
operator&, [595](#)
operator|, [630](#)
partial_sum, [631](#)
put_money, [632](#)
put_time, [632](#)
quoted, [632](#)
rbegin, [632](#), [633](#)
ref, [633](#), [634](#)
rend, [634](#), [635](#)
replace_copy, [635](#)
resetiosflags, [635](#)
return_temporary_buffer, [636](#)
right, [636](#)
round_to_nearest, [562](#)
round_toward_infinity, [562](#)
round_toward_neg_infinity, [562](#)
round_toward_zero, [562](#)
scientific, [636](#)
set_new_handler, [636](#)
setbase, [636](#)
setfill, [637](#)
setiosflags, [637](#)
setprecision, [637](#)
setw, [637](#)
showbase, [638](#)
showpoint, [638](#)
showpos, [638](#)
skipws, [638](#)
static_pointer_cast, [638](#)
streamoff, [560](#)
streampos, [560](#)
streamsize, [561](#)
swap, [639–641](#)
tolower, [641](#)
toupper, [642](#)
try_lock, [642](#)
u16streampos, [561](#)
u32streampos, [561](#)
uninitialized_copy, [642](#)
uninitialized_copy_n, [643](#)
uninitialized_fill, [643](#)
uninitialized_fill_n, [644](#)
unitbuf, [644](#)
uppercase, [644](#)
wcerr, [647](#)
wcin, [647](#)
wclog, [647](#)
wcout, [647](#)
ws, [644](#)
wstreampos, [561](#)
std::__allocated_ptr
 __allocated_ptr, [1424](#)
 ~__allocated_ptr, [1424](#)
get, [1425](#)
operator=, [1425](#)

std::__allocated_ptr< _Alloc >, 1424
 std::__atomic_base< _ITp >, 1425
 std::__atomic_base< _PTp * >, 1426
 std::__atomic_flag_base, 1427
 std::__basic_future
 _M_get_result, 1429
 _Ptr, 1429
 std::__basic_future< _Res >, 1428
 std::__codecvt_abstract_base
 do_out, 1431
 in, 1431
 out, 1432
 unshift, 1433
 std::__codecvt_abstract_base< _InternT, _ExternT, _↔
 StateT >, 1430
 std::__ctype_abstract_base
 char_type, 1436
 do_is, 1436, 1437
 do_narrow, 1437, 1438
 do_scan_is, 1438
 do_scan_not, 1439
 do_tolower, 1439, 1440
 do_toupper, 1440, 1441
 do_widen, 1441, 1442
 is, 1442, 1443
 narrow, 1443, 1444
 scan_is, 1444
 scan_not, 1445
 tolower, 1445, 1446
 toupper, 1446, 1447
 widen, 1447, 1448
 std::__ctype_abstract_base< _CharT >, 1434
 std::__debug, 647
 operator<=, 652
 operator>, 652
 operator>=, 652
 swap, 652
 std::__debug::bitset< _Nb >, 1448
 std::__debug::deque
 _M_attach, 1453
 _M_attach_single, 1453
 _M_const_iterators, 1454
 _M_detach, 1453
 _M_detach_all, 1453
 _M_detach_single, 1453
 _M_detach_singular, 1453
 _M_get_mutex, 1453
 _M_invalidate_all, 1453
 _M_invalidate_if, 1453
 _M_iterators, 1454
 _M_revalidate_singular, 1454
 _M_swap, 1454
 _M_transfer_from_if, 1454
 _M_version, 1454
 std::__debug::deque< _Tp, _Allocator >, 1450
 std::__debug::forward_list
 _M_attach, 1457
 _M_attach_single, 1457
 _M_const_iterators, 1459
 _M_detach, 1457
 _M_detach_all, 1457
 _M_detach_single, 1457
 _M_detach_singular, 1458
 _M_get_mutex, 1458
 _M_invalidate_all, 1458
 _M_invalidate_if, 1458
 _M_iterators, 1459
 _M_revalidate_singular, 1458
 _M_transfer_from_if, 1458
 _M_version, 1459
 std::__debug::forward_list< _Tp, _Alloc >, 1455
 std::__debug::list
 _M_attach, 1462
 _M_attach_single, 1462
 _M_const_iterators, 1463
 _M_detach, 1462
 _M_detach_all, 1462
 _M_detach_single, 1462
 _M_detach_singular, 1462
 _M_get_mutex, 1462
 _M_invalidate_all, 1463
 _M_invalidate_if, 1463
 _M_iterators, 1463
 _M_revalidate_singular, 1463
 _M_swap, 1463
 _M_transfer_from_if, 1463
 _M_version, 1464
 std::__debug::list< _Tp, _Allocator >, 1459
 std::__debug::map
 _M_attach, 1467
 _M_attach_single, 1467
 _M_const_iterators, 1468
 _M_detach, 1467
 _M_detach_all, 1467
 _M_detach_single, 1467
 _M_detach_singular, 1467
 _M_get_mutex, 1467
 _M_invalidate_all, 1468
 _M_invalidate_if, 1468
 _M_iterators, 1468
 _M_revalidate_singular, 1468
 _M_swap, 1468
 _M_transfer_from_if, 1468
 _M_version, 1469
 std::__debug::map< _Key, _Tp, _Compare, _Allocator >, 1464
 std::__debug::multimap
 _M_attach, 1472

- [_M_attach_single](#), 1472
- [_M_const_iterators](#), 1473
- [_M_detach](#), 1472
- [_M_detach_all](#), 1472
- [_M_detach_single](#), 1472
- [_M_detach_singular](#), 1472
- [_M_get_mutex](#), 1472
- [_M_invalidate_all](#), 1473
- [_M_invalidate_if](#), 1473
- [_M_iterators](#), 1473
- [_M_revalidate_singular](#), 1473
- [_M_swap](#), 1473
- [_M_transfer_from_if](#), 1473
- [_M_version](#), 1474
- [std::__debug::multimap< _Key, _Tp, _Compare, _Alloc, _Allocator >](#), 1469
- [std::__debug::multiset](#)
 - [_M_attach](#), 1477
 - [_M_attach_single](#), 1477
 - [_M_const_iterators](#), 1478
 - [_M_detach](#), 1477
 - [_M_detach_all](#), 1477
 - [_M_detach_single](#), 1477
 - [_M_detach_singular](#), 1477
 - [_M_get_mutex](#), 1477
 - [_M_invalidate_all](#), 1478
 - [_M_invalidate_if](#), 1478
 - [_M_iterators](#), 1478
 - [_M_revalidate_singular](#), 1478
 - [_M_swap](#), 1478
 - [_M_transfer_from_if](#), 1478
 - [_M_version](#), 1479
- [std::__debug::multiset< _Key, _Compare, _Allocator >](#), 1474
- [std::__debug::set](#)
 - [_M_attach](#), 1482
 - [_M_attach_single](#), 1482
 - [_M_const_iterators](#), 1483
 - [_M_detach](#), 1482
 - [_M_detach_all](#), 1482
 - [_M_detach_single](#), 1482
 - [_M_detach_singular](#), 1482
 - [_M_get_mutex](#), 1482
 - [_M_invalidate_all](#), 1483
 - [_M_invalidate_if](#), 1483
 - [_M_iterators](#), 1483
 - [_M_revalidate_singular](#), 1483
 - [_M_swap](#), 1483
 - [_M_transfer_from_if](#), 1483
 - [_M_version](#), 1484
- [std::__debug::set< _Key, _Compare, _Allocator >](#), 1479
- [std::__debug::unordered_map](#)
 - [_M_attach](#), 1487
 - [_M_attach_local](#), 1487
 - [_M_attach_local_single](#), 1487
 - [_M_attach_single](#), 1487
 - [_M_const_iterators](#), 1489
 - [_M_const_local_iterators](#), 1489
 - [_M_detach](#), 1487
 - [_M_detach_all](#), 1487
 - [_M_detach_local](#), 1487
 - [_M_detach_local_single](#), 1487
 - [_M_detach_single](#), 1488
 - [_M_detach_singular](#), 1488
 - [_M_get_mutex](#), 1488
 - [_M_invalidate_all](#), 1488
 - [_M_invalidate_if](#), 1488
 - [_M_invalidate_local_if](#), 1488
 - [_M_iterators](#), 1489
 - [_M_local_iterators](#), 1489
 - [_M_revalidate_singular](#), 1488
 - [_M_swap](#), 1489
 - [_M_version](#), 1489
- [std::__debug::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >](#), 1484
- [std::__debug::unordered_multimap](#)
 - [_M_attach](#), 1493
 - [_M_attach_local](#), 1493
 - [_M_attach_local_single](#), 1493
 - [_M_attach_single](#), 1493
 - [_M_const_iterators](#), 1495
 - [_M_const_local_iterators](#), 1495
 - [_M_detach](#), 1493
 - [_M_detach_all](#), 1493
 - [_M_detach_local](#), 1493
 - [_M_detach_local_single](#), 1493
 - [_M_detach_single](#), 1493
 - [_M_detach_singular](#), 1493
 - [_M_get_mutex](#), 1494
 - [_M_invalidate_all](#), 1494
 - [_M_invalidate_if](#), 1494
 - [_M_invalidate_local_if](#), 1494
 - [_M_iterators](#), 1495
 - [_M_local_iterators](#), 1495
 - [_M_revalidate_singular](#), 1494
 - [_M_swap](#), 1494
 - [_M_version](#), 1495
- [std::__debug::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >](#), 1490
- [std::__debug::unordered_multiset](#)
 - [_M_attach](#), 1498
 - [_M_attach_local](#), 1498
 - [_M_attach_local_single](#), 1498
 - [_M_attach_single](#), 1499
 - [_M_const_iterators](#), 1500
 - [_M_const_local_iterators](#), 1500
 - [_M_detach](#), 1499
 - [_M_detach_all](#), 1499

- [_M_detach_local, 1499](#)
- [_M_detach_local_single, 1499](#)
- [_M_detach_single, 1499](#)
- [_M_detach_singular, 1499](#)
- [_M_get_mutex, 1499](#)
- [_M_invalidate_all, 1499](#)
- [_M_invalidate_if, 1500](#)
- [_M_invalidate_local_if, 1500](#)
- [_M_iterators, 1501](#)
- [_M_local_iterators, 1501](#)
- [_M_revalidate_singular, 1500](#)
- [_M_swap, 1500](#)
- [_M_version, 1501](#)
- [std::__debug::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, 1496](#)
- [std::__debug::unordered_set](#)
 - [_M_attach, 1504](#)
 - [_M_attach_local, 1504](#)
 - [_M_attach_local_single, 1504](#)
 - [_M_attach_single, 1504](#)
 - [_M_const_iterators, 1506](#)
 - [_M_const_local_iterators, 1506](#)
 - [_M_detach, 1504](#)
 - [_M_detach_all, 1504](#)
 - [_M_detach_local, 1504](#)
 - [_M_detach_local_single, 1504](#)
 - [_M_detach_single, 1505](#)
 - [_M_detach_singular, 1505](#)
 - [_M_get_mutex, 1505](#)
 - [_M_invalidate_all, 1505](#)
 - [_M_invalidate_if, 1505](#)
 - [_M_invalidate_local_if, 1505](#)
 - [_M_iterators, 1506](#)
 - [_M_local_iterators, 1506](#)
 - [_M_revalidate_singular, 1505](#)
 - [_M_swap, 1506](#)
 - [_M_version, 1506](#)
- [std::__debug::unordered_set< _Value, _Hash, _Pred, _Alloc >, 1501](#)
- [std::__debug::vector](#)
 - [_M_attach, 1510](#)
 - [_M_attach_single, 1510](#)
 - [_M_const_iterators, 1512](#)
 - [_M_detach, 1510](#)
 - [_M_detach_all, 1510](#)
 - [_M_detach_single, 1510](#)
 - [_M_detach_singular, 1510](#)
 - [_M_get_mutex, 1511](#)
 - [_M_invalidate_all, 1511](#)
 - [_M_invalidate_if, 1511](#)
 - [_M_iterators, 1512](#)
 - [_M_revalidate_singular, 1511](#)
 - [_M_swap, 1511](#)
 - [_M_transfer_from_if, 1511](#)
 - [_M_version, 1512](#)
- [vector, 1510](#)
- [std::__debug::vector< _Tp, _Allocator >, 1507](#)
- [std::__detail, 653](#)
 - [operator<<, 655](#)
 - [operator>>, 656](#)
- [std::__detail::__BracketMatcher< _TraitsT, __icase, __collate >, 1512](#)
- [std::__detail::__Compiler< _TraitsT >, 1513](#)
- [std::__detail::__Default_ranged_hash, 1514](#)
- [std::__detail::__Equal_helper< _Key, _Value, _ExtractKey, _Equal, _HashCodeType, __cache_hash_code >, 1514](#)
- [std::__detail::__Equal_helper< _Key, _Value, _ExtractKey, _Equal, _HashCodeType, false >, 1514](#)
- [std::__detail::__Equal_helper< _Key, _Value, _ExtractKey, _Equal, _HashCodeType, true >, 1515](#)
- [std::__detail::__Equality< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, false >, 1516](#)
- [std::__detail::__Equality< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, true >, 1517](#)
- [std::__detail::__Equality< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits, Unique_keys >, 1515](#)
- [std::__detail::__Equality_base, 1518](#)
- [std::__detail::__Executor< _Bilter, _Alloc, _TraitsT, __dfs_mode >, 1518](#)
- [std::__detail::__Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Default_ranged_hash, false >, 1520](#)
- [std::__detail::__Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Default_ranged_hash, true >, 1522](#)
- [std::__detail::__Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, __cache_hash_code >, 1520](#)
- [std::__detail::__Hash_code_base< _Key, _Value, _ExtractKey, _H1, _H2, _Hash, false >, 1523](#)
- [std::__detail::__Hash_node< _Value, _Cache_hash_code >, 1524](#)
- [std::__detail::__Hash_node< _Value, false >, 1525](#)
- [std::__detail::__Hash_node< _Value, true >, 1526](#)
- [std::__detail::__Hash_node_base, 1527](#)
- [std::__detail::__Hash_node_value_base< _Value >, 1528](#)
- [std::__detail::__Hashtable_alloc< _NodeAlloc >, 1529](#)
- [std::__detail::__Hashtable_base< _Key, _Value, _ExtractKey, _Equal, _H1, _H2, _Hash, _Traits >, 1530](#)
- [std::__detail::__Hashtable_ebo_helper< _Nm, _Tp, __use_ebo >, 1532](#)
- [std::__detail::__Hashtable_ebo_helper< _Nm, _Tp, false >, 1532](#)
- [std::__detail::__Hashtable_ebo_helper< _Nm, _Tp, true >, 1532](#)

- 1532
- std::__detail::_Hashtable_traits< _Cache_hash_code, _←
_Constant_iterators, _Unique_keys >, 1533
- std::__detail::_Insert< _Key, _Value, _Alloc, _ExtractKey,
_Equal, _H1, _H2, _Hash, _RehashPolicy, _←
Traits, false, _Unique_keys >, 1535
- std::__detail::_Insert< _Key, _Value, _Alloc, _ExtractKey,
_Equal, _H1, _H2, _Hash, _RehashPolicy, _←
Traits, true, false >, 1537
- std::__detail::_Insert< _Key, _Value, _Alloc, _ExtractKey,
_Equal, _H1, _H2, _Hash, _RehashPolicy, _←
Traits, true, true >, 1538
- std::__detail::_Insert< _Key, _Value, _Alloc, _ExtractKey,
_Equal, _H1, _H2, _Hash, _RehashPolicy, _←
_Traits, _Constant_iterators, _Unique_keys >,
1534
- std::__detail::_Insert_base< _Key, _Value, _Alloc, _←
_ExtractKey, _Equal, _H1, _H2, _Hash, _←
RehashPolicy, _Traits >, 1540
- std::__detail::_List_node_base, 1541
- std::__detail::_Local_const_iterator< _Key, _Value, _←
ExtractKey, _H1, _H2, _Hash, __constant_←
iterators, __cache >, 1542
- std::__detail::_Local_iterator< _Key, _Value, _ExtractKey,
_H1, _H2, _Hash, __constant_iterators, __←
cache >, 1543
- std::__detail::_Local_iterator_base< _Key, _Value, _←
ExtractKey, _H1, _H2, _Hash, __cache_hash←
_code >, 1544
- std::__detail::_Local_iterator_base< _Key, _Value, _←
ExtractKey, _H1, _H2, _Hash, true >, 1545
- std::__detail::_Map_base< _Key, _Pair, _Alloc, _←
Select1st, _Equal, _H1, _H2, _Hash, _←
RehashPolicy, _Traits, false >, 1547
- std::__detail::_Map_base< _Key, _Pair, _Alloc, _←
Select1st, _Equal, _H1, _H2, _Hash, _←
RehashPolicy, _Traits, true >, 1547
- std::__detail::_Map_base< _Key, _Value, _Alloc, _←
ExtractKey, _Equal, _H1, _H2, _Hash, _←
RehashPolicy, _Traits, _Unique_keys >, 1546
- std::__detail::_Mod_range_hashing, 1548
- std::__detail::_Node_const_iterator< _Value, __←
constant_iterators, __cache >, 1548
- std::__detail::_Node_iterator< _Value, __constant_←
iterators, __cache >, 1550
- std::__detail::_Node_iterator_base< _Value, _Cache_←
hash_code >, 1551
- std::__detail::_Prime_rehash_policy, 1552
- std::__detail::_Quoted_string< _String, _CharT >, 1552
- std::__detail::_Rehash_base< _Key, _Value, _Alloc, _←
ExtractKey, _Equal, _H1, _H2, _Hash, _Prime_←
rehash_policy, _Traits >, 1554
- std::__detail::_Rehash_base< _Key, _Value, _Alloc, _←
_ExtractKey, _Equal, _H1, _H2, _Hash, _←
RehashPolicy, _Traits >, 1553
- std::__detail::_Scanner
_TokenT, 1556
- std::__detail::_Scanner< _CharT >, 1555
- std::__detail::_StateSeq< _TraitsT >, 1556
- std::__exception_ptr::exception_ptr, 1557
- std::__future_base, 1558
_Ptr, 1559
- std::__future_base::_Result< _Res >, 1559
- std::__future_base::_Result< _Res & >, 1560
- std::__future_base::_Result< void >, 1561
- std::__future_base::_Result_alloc< _Res, _Alloc >, 1562
- std::__future_base::_Result_base, 1563
- std::__is_location_invariant< _Tp >, 1564
- std::__is_nullptr_t< _Tp >, 1565
- std::__is_tuple_like_impl< std::pair< _T1, _T2 > >, 1566
- std::__iterator_traits< _Iterator, typename >, 1567
- std::__numeric_limits_base, 1567
digits, 1568
digits10, 1568
has_denorm, 1569
has_denorm_loss, 1569
has_infinity, 1569
has_quiet_NaN, 1569
has_signaling_NaN, 1569
is_bounded, 1569
is_exact, 1569
is_iec559, 1569
is_integer, 1570
is_modulo, 1570
is_signed, 1570
is_specialized, 1570
max_digits10, 1570
max_exponent, 1570
max_exponent10, 1570
min_exponent, 1571
min_exponent10, 1571
radix, 1571
round_style, 1571
tinyness_before, 1571
traps, 1571
- std::__parallel, 656
- std::__parallel::_CRandNumber< _MustBeInt >, 1572
- std::__profile, 674
operator<=, 679
operator>, 679
operator>=, 679
swap, 679
- std::__profile::bitset< _Nb >, 1572
- std::__profile::deque< _Tp, _Allocator >, 1573
- std::__profile::forward_list< _Tp, _Alloc >, 1574
- std::__profile::list< _Tp, _Allocator >, 1575
- std::__profile::map< _Key, _Tp, _Compare, _Allocator >,
1577

std::__profile::multimap< _Key, _Tp, _Compare, _Allocator >, 1580
 std::__profile::multiset< _Key, _Compare, _Allocator >, 1583
 std::__profile::set< _Key, _Compare, _Allocator >, 1585
 std::__profile::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, 1588
 std::__profile::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, 1590
 std::__profile::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, 1592
 std::__profile::unordered_set< _Key, _Hash, _Pred, _Alloc >, 1594
 std::__shared_mutex_cv, 1596
 std::_Base_bitset
 _M_w, 1598
 std::_Base_bitset< 0 >, 1598
 std::_Base_bitset< 1 >, 1599
 std::_Base_bitset< _Nw >, 1596
 std::_Bind< _Signature >, 1600
 std::_Bind_result< _Result, _Signature >, 1601
 std::_Deque_base
 _M_initialize_map, 1602
 std::_Deque_base< _Tp, _Alloc >, 1601
 std::_Deque_iterator
 _M_set_node, 1604
 std::_Deque_iterator< _Tp, _Ref, _Ptr >, 1603
 std::_Enable_copy_move< _Copy, _CopyAssignment, _Move, _MoveAssignment, _Tag >, 1605
 std::_Enable_default_constructor< _Switch, _Tag >, 1605
 std::_Enable_destructor< _Switch, _Tag >, 1606
 std::_Enable_special_members< _Default, _Destructor, _Copy, _CopyAssignment, _Move, _MoveAssignment, _Tag >, 1606
 std::_Function_base, 1607
 std::_Fwd_list_base< _Tp, _Alloc >, 1608
 std::_Fwd_list_const_iterator< _Tp >, 1609
 std::_Fwd_list_iterator< _Tp >, 1610
 std::_Fwd_list_node< _Tp >, 1611
 std::_Fwd_list_node_base, 1612
 std::_Hashtable< _Key, _Value, _Alloc, _ExtractKey, _Equal, _H1, _H2, _Hash, _RehashPolicy, _Traits >, 1613
 std::_List_base< _Tp, _Alloc >, 1619
 std::_List_const_iterator< _Tp >, 1620
 std::_List_iterator< _Tp >, 1621
 std::_List_node< _Tp >, 1622
 std::_Maybe_get_result_type< _Functor, typename >, 1623
 std::_Maybe_unary_or_binary_function< _Res, _ArgTypes >, 1624
 std::_Maybe_unary_or_binary_function< _Res, _T1 >, 1624
 argument_type, 1625
 result_type, 1625
 std::_Maybe_unary_or_binary_function< _Res, _T1, _T2 >, 1625
 first_argument_type, 1626
 result_type, 1626
 second_argument_type, 1626
 std::_Maybe_wrap_member_pointer< _Tp >, 1626
 std::_Maybe_wrap_member_pointer< _Tp _Class::* >, 1627
 std::_Mu< _Arg, _IsBindExp, _IsPlaceholder >, 1628
 std::_Mu< _Arg, false, false >, 1628
 std::_Mu< _Arg, false, true >, 1628
 std::_Mu< _Arg, true, false >, 1629
 std::_Mu< reference_wrapper< _Tp >, false, false >, 1629
 std::_Placeholder< _Num >, 1630
 std::_Reference_wrapper_base< _Tp >, 1630
 std::_Reference_wrapper_base_impl< _Unary, _Binary, _Tp >, 1631
 std::_Sp_ebo_helper< _Nm, _Tp, false >, 1631
 std::_Sp_ebo_helper< _Nm, _Tp, true >, 1632
 std::_Temporary_buffer
 _Temporary_buffer, 1634
 begin, 1634
 end, 1634
 requested_size, 1634
 size, 1634
 std::_Temporary_buffer< _ForwardIterator, _Tp >, 1633
 std::_Tuple_impl< _Idx, _Elements >, 1635
 std::_Tuple_impl< _Idx, _Head, _Tail... >, 1635
 std::_V2::condition_variable_any, 1637
 std::_V2::error_category, 1638
 std::_Vector_base< _Tp, _Alloc >, 1638
 std::_Weak_result_type< _Functor >, 1640
 std::_Weak_result_type_impl< _Functor >, 1641
 std::_Weak_result_type_impl< _Res(*)(_ArgTypes...) >, 1642
 std::_Weak_result_type_impl< _Res(&)(_ArgTypes...) >, 1641
 std::_Weak_result_type_impl< _Res(_ArgTypes...) >, 1642
 std::_Weak_result_type_impl< _Res(_Class::*)(_ArgTypes...) const >, 1642
 std::_Weak_result_type_impl< _Res(_Class::*)(_ArgTypes...) const volatile >, 1643
 std::_Weak_result_type_impl< _Res(_Class::*)(_ArgTypes...) volatile >, 1643
 std::_Weak_result_type_impl< _Res(_Class::*)(_ArgTypes...) >, 1644
 std::adopt_lock_t, 1644
 std::allocator< _Tp >, 1645
 std::allocator< void >, 1647
 std::allocator_arg_t, 1648
 std::allocator_traits

- allocate, 1651, 1652
- allocator_type, 1649
- const_pointer, 1649
- const_void_pointer, 1650
- construct, 1652
- deallocate, 1652
- destroy, 1653
- difference_type, 1650
- is_always_equal, 1650
- max_size, 1653
- pointer, 1650
- propagate_on_container_copy_assignment, 1650
- propagate_on_container_move_assignment, 1651
- propagate_on_container_swap, 1651
- select_on_container_copy_construction, 1654
- size_type, 1651
- value_type, 1651
- void_pointer, 1651
- std::allocator_traits< _Alloc >, 1648
- std::allocator_traits< allocator< _Tp > >, 1654
 - allocate, 1656, 1657
 - allocator_type, 1655
 - const_pointer, 1655
 - const_void_pointer, 1655
 - construct, 1657
 - deallocate, 1657
 - destroy, 1658
 - difference_type, 1655
 - is_always_equal, 1655
 - max_size, 1658
 - pointer, 1656
 - propagate_on_container_copy_assignment, 1656
 - propagate_on_container_move_assignment, 1656
 - propagate_on_container_swap, 1656
 - select_on_container_copy_construction, 1658
 - size_type, 1656
 - value_type, 1656
 - void_pointer, 1656
- std::array< _Tp, _Nm >, 1659
- std::atomic< _Tp >, 1661
- std::atomic< _Tp * >, 1662
- std::atomic< bool >, 1663
- std::atomic< char >, 1664
- std::atomic< char16_t >, 1666
- std::atomic< char32_t >, 1667
- std::atomic< int >, 1668
- std::atomic< long >, 1670
- std::atomic< long long >, 1671
- std::atomic< short >, 1672
- std::atomic< signed char >, 1674
- std::atomic< unsigned char >, 1675
- std::atomic< unsigned int >, 1676
- std::atomic< unsigned long >, 1678
- std::atomic< unsigned long long >, 1679
- std::atomic< unsigned short >, 1680
- std::atomic< wchar_t >, 1682
- std::atomic_flag, 1683
- std::auto_ptr
 - ~auto_ptr, 1686
 - auto_ptr, 1685, 1686
 - element_type, 1685
 - get, 1687
 - operator*, 1687
 - operator->, 1687
 - operator=, 1687, 1688
 - release, 1688
 - reset, 1688
- std::auto_ptr< _Tp >, 1684
- std::auto_ptr_ref< _Tp1 >, 1689
- std::back_insert_iterator
 - back_insert_iterator, 1692
 - container_type, 1691
 - difference_type, 1691
 - iterator_category, 1691
 - operator*, 1692
 - operator++, 1692
 - operator=, 1692
 - pointer, 1691
 - reference, 1691
 - value_type, 1691
- std::back_insert_iterator< _Container >, 1690
- std::bad_alloc, 1693
 - what, 1694
- std::bad_cast, 1694
 - what, 1695
- std::bad_exception, 1695
 - what, 1696
- std::bad_function_call, 1696
 - what, 1696
- std::bad_typeid, 1697
 - what, 1697
- std::bad_weak_ptr, 1698
 - what, 1698
- std::basic_filebuf
 - _M_buf, 1721
 - _M_buf_locale, 1721
 - _M_buf_size, 1721
 - _M_create_pback, 1703
 - _M_destroy_pback, 1703
 - _M_ext_buf, 1721
 - _M_ext_buf_size, 1722
 - _M_ext_next, 1722
 - _M_in_beg, 1722
 - _M_in_cur, 1722
 - _M_in_end, 1722
 - _M_mode, 1722
 - _M_out_beg, 1723
 - _M_out_cur, 1723

- [_M_out_end, 1723](#)
- [_M_pback, 1723](#)
- [_M_pback_cur_save, 1723](#)
- [_M_pback_end_save, 1724](#)
- [_M_pback_init, 1724](#)
- [_M_reading, 1724](#)
- [_M_set_buffer, 1703](#)
- [~basic_filebuf, 1702](#)
- [basic_filebuf, 1702](#)
- [close, 1703](#)
- [eback, 1703](#)
- [egptr, 1704](#)
- [epptr, 1704](#)
- [gbump, 1704](#)
- [getloc, 1705](#)
- [gptr, 1705](#)
- [imbue, 1705](#)
- [in_avail, 1706](#)
- [is_open, 1706](#)
- [open, 1706, 1707](#)
- [overflow, 1708](#)
- [pbackfail, 1708](#)
- [pbase, 1710](#)
- [pbump, 1710](#)
- [pptr, 1711](#)
- [pubimbue, 1711](#)
- [pubseekoff, 1711](#)
- [pubseekpos, 1712](#)
- [pubsetbuf, 1712](#)
- [pubsync, 1712](#)
- [sbumpc, 1712](#)
- [seekoff, 1713](#)
- [seekpos, 1713](#)
- [setbuf, 1713](#)
- [setg, 1714](#)
- [setp, 1714](#)
- [sgetc, 1715](#)
- [sgetn, 1715](#)
- [showmanyc, 1716](#)
- [snextc, 1716](#)
- [sputbackc, 1716](#)
- [sputc, 1717](#)
- [sputn, 1717](#)
- [sungetc, 1718](#)
- [sync, 1718](#)
- [uflow, 1718](#)
- [underflow, 1719](#)
- [xsgetn, 1719](#)
- [xsputn, 1720](#)
- [std::basic_filebuf<_CharT, _Traits>, 1699](#)
- [std::basic_fstream](#)
 - [_M_gcount, 1777](#)
 - [_M_getloc, 1735](#)
 - [_M_write, 1735](#)
 - [__num_put_type, 1732](#)
 - [~basic_fstream, 1735](#)
 - [adjustfield, 1777](#)
 - [app, 1778](#)
 - [ate, 1778](#)
 - [bad, 1736](#)
 - [badbit, 1778](#)
 - [basefield, 1778](#)
 - [basic_fstream, 1734, 1735](#)
 - [beg, 1778](#)
 - [binary, 1779](#)
 - [boolalpha, 1779](#)
 - [clear, 1736](#)
 - [close, 1736](#)
 - [copyfmt, 1736](#)
 - [cur, 1779](#)
 - [dec, 1779](#)
 - [end, 1779](#)
 - [eof, 1737](#)
 - [eofbit, 1780](#)
 - [event, 1734](#)
 - [event_callback, 1732](#)
 - [exceptions, 1737](#)
 - [fail, 1738](#)
 - [failbit, 1780](#)
 - [fill, 1738](#)
 - [fixed, 1780](#)
 - [flags, 1739](#)
 - [floatfield, 1780](#)
 - [flush, 1740](#)
 - [fmtflags, 1732](#)
 - [gcount, 1740](#)
 - [get, 1740–1743](#)
 - [getline, 1743, 1744](#)
 - [getloc, 1744](#)
 - [good, 1745](#)
 - [goodbit, 1781](#)
 - [hex, 1781](#)
 - [ignore, 1745, 1746](#)
 - [imbue, 1746](#)
 - [in, 1781](#)
 - [init, 1747](#)
 - [internal, 1781](#)
 - [iostate, 1733](#)
 - [is_open, 1747](#)
 - [iword, 1747](#)
 - [left, 1782](#)
 - [narrow, 1748](#)
 - [oct, 1782](#)
 - [open, 1748, 1749](#)
 - [openmode, 1733](#)
 - [operator bool, 1749](#)
 - [operator!, 1749](#)
 - [operator<<, 1749–1755](#)

- operator>>, 1756–1758, 1760–1763
- out, 1782
- peek, 1763
- precision, 1763, 1764
- put, 1764
- putback, 1765
- pword, 1765
- rdbuf, 1766
- rdstate, 1766
- read, 1767
- readsome, 1767
- register_callback, 1768
- right, 1782
- scientific, 1782
- seekdir, 1733
- seekg, 1768, 1769
- seekp, 1770
- setf, 1771
- setstate, 1771
- showbase, 1783
- showpoint, 1783
- showpos, 1783
- skipws, 1783
- sync, 1772
- sync_with_stdio, 1772
- tellg, 1773
- tellp, 1773
- tie, 1773, 1774
- trunc, 1783
- unget, 1774
- unitbuf, 1783
- unsetf, 1775
- uppercase, 1784
- widen, 1775
- width, 1776
- write, 1776
- xalloc, 1777
- std::basic_fstream<_CharT, _Traits >, 1725
- std::basic_ifstream
 - _M_gcount, 1826
 - _M_getloc, 1793
 - __num_put_type, 1790
 - ~basic_ifstream, 1793
 - adjustfield, 1826
 - app, 1826
 - ate, 1826
 - bad, 1793
 - badbit, 1827
 - basefield, 1827
 - basic_ifstream, 1792, 1793
 - beg, 1827
 - binary, 1827
 - boolalpha, 1827
 - clear, 1794
 - close, 1794
 - copyfmt, 1794
 - cur, 1828
 - dec, 1828
 - end, 1828
 - eof, 1795
 - eofbit, 1828
 - event, 1792
 - event_callback, 1790
 - exceptions, 1795
 - fail, 1796
 - failbit, 1828
 - fill, 1796
 - fixed, 1829
 - flags, 1797
 - floatfield, 1829
 - fmtflags, 1790
 - gcount, 1798
 - get, 1798–1800
 - getline, 1801, 1802
 - getloc, 1802
 - good, 1802
 - goodbit, 1829
 - hex, 1829
 - ignore, 1802, 1803
 - imbue, 1804
 - in, 1830
 - init, 1804
 - internal, 1830
 - iostate, 1791
 - is_open, 1804
 - iword, 1805
 - left, 1830
 - narrow, 1805
 - oct, 1830
 - open, 1806
 - openmode, 1791
 - operator bool, 1806
 - operator!, 1806
 - operator>>, 1807–1809, 1811–1814
 - out, 1830
 - peek, 1814
 - precision, 1814, 1815
 - putback, 1815
 - pword, 1816
 - rdbuf, 1816, 1817
 - rdstate, 1817
 - read, 1817
 - readsome, 1818
 - register_callback, 1819
 - right, 1831
 - scientific, 1831
 - seekdir, 1792
 - seekg, 1819, 1820

- setf, [1820](#), [1821](#)
- setstate, [1821](#)
- showbase, [1831](#)
- showpoint, [1831](#)
- showpos, [1831](#)
- skipws, [1831](#)
- sync, [1821](#)
- sync_with_stdio, [1822](#)
- tellg, [1822](#)
- tie, [1823](#)
- trunc, [1832](#)
- unget, [1823](#)
- unitbuf, [1832](#)
- unsetf, [1824](#)
- uppercase, [1832](#)
- widen, [1824](#)
- width, [1825](#)
- xalloc, [1826](#)
- std::basic_ifstream< _CharT, _Traits >, [1784](#)
- std::basic_ios
 - _M_getloc, [1840](#)
 - __ctype_type, [1836](#)
 - __num_get_type, [1836](#)
 - __num_put_type, [1836](#)
 - ~basic_ios, [1839](#)
 - adjustfield, [1854](#)
 - app, [1854](#)
 - ate, [1854](#)
 - bad, [1840](#)
 - badbit, [1854](#)
 - basefield, [1854](#)
 - basic_ios, [1839](#), [1840](#)
 - beg, [1854](#)
 - binary, [1855](#)
 - boolalpha, [1855](#)
 - char_type, [1836](#)
 - clear, [1840](#)
 - copyfmt, [1841](#)
 - cur, [1855](#)
 - dec, [1855](#)
 - end, [1855](#)
 - eof, [1841](#)
 - eofbit, [1856](#)
 - event, [1839](#)
 - event_callback, [1836](#)
 - exceptions, [1841](#), [1842](#)
 - fail, [1842](#)
 - failbit, [1856](#)
 - fill, [1843](#)
 - fixed, [1856](#)
 - flags, [1843](#), [1844](#)
 - floatfield, [1856](#)
 - fmtflags, [1837](#)
 - getloc, [1844](#)
 - good, [1844](#)
 - goodbit, [1857](#)
 - hex, [1857](#)
 - imbue, [1845](#)
 - in, [1857](#)
 - init, [1845](#)
 - int_type, [1837](#)
 - internal, [1857](#)
 - iostate, [1838](#)
 - isword, [1845](#)
 - left, [1858](#)
 - narrow, [1846](#)
 - oct, [1858](#)
 - off_type, [1838](#)
 - openmode, [1838](#)
 - operator bool, [1846](#)
 - operator!, [1846](#)
 - out, [1858](#)
 - pos_type, [1838](#)
 - precision, [1847](#)
 - pword, [1847](#)
 - rdbuf, [1848](#)
 - rdstate, [1849](#)
 - register_callback, [1849](#)
 - right, [1858](#)
 - scientific, [1858](#)
 - seekdir, [1839](#)
 - setf, [1849](#), [1850](#)
 - setstate, [1850](#)
 - showbase, [1859](#)
 - showpoint, [1859](#)
 - showpos, [1859](#)
 - skipws, [1859](#)
 - sync_with_stdio, [1851](#)
 - tie, [1851](#)
 - traits_type, [1839](#)
 - trunc, [1859](#)
 - unitbuf, [1859](#)
 - unsetf, [1852](#)
 - uppercase, [1860](#)
 - widen, [1852](#)
 - width, [1852](#), [1853](#)
 - xalloc, [1853](#)
- std::basic_ios< _CharT, _Traits >, [1832](#)
- std::basic_iostream
 - _M_gcount, [1913](#)
 - _M_getloc, [1870](#)
 - _M_write, [1870](#)
 - __num_put_type, [1867](#)
 - ~basic_iostream, [1869](#)
 - adjustfield, [1913](#)
 - app, [1914](#)
 - ate, [1914](#)
 - bad, [1870](#)

- badbit, [1914](#)
- basefield, [1914](#)
- basic_istream, [1869](#)
- beg, [1914](#)
- binary, [1915](#)
- boolalpha, [1915](#)
- clear, [1870](#)
- copyfmt, [1872](#)
- cur, [1915](#)
- dec, [1915](#)
- end, [1915](#)
- eof, [1872](#)
- eofbit, [1916](#)
- event, [1869](#)
- event_callback, [1867](#)
- exceptions, [1872](#), [1873](#)
- fail, [1873](#)
- failbit, [1916](#)
- fill, [1874](#)
- fixed, [1916](#)
- flags, [1874](#), [1875](#)
- floatfield, [1916](#)
- flush, [1875](#)
- fmtflags, [1867](#)
- gcount, [1875](#)
- get, [1876](#)–[1878](#)
- getline, [1879](#), [1880](#)
- getloc, [1880](#)
- good, [1880](#)
- goodbit, [1917](#)
- hex, [1917](#)
- ignore, [1880](#), [1881](#)
- imbue, [1882](#)
- in, [1917](#)
- init, [1882](#)
- internal, [1917](#)
- iostate, [1868](#)
- isword, [1882](#)
- left, [1918](#)
- narrow, [1883](#)
- oct, [1918](#)
- openmode, [1868](#)
- operator bool, [1883](#)
- operator!, [1884](#)
- operator<<, [1884](#)–[1887](#), [1889](#), [1890](#)
- operator>>, [1892](#)–[1894](#), [1896](#)–[1899](#)
- out, [1918](#)
- peek, [1899](#)
- precision, [1899](#), [1900](#)
- put, [1900](#)
- putback, [1901](#)
- pword, [1901](#)
- rdbuf, [1902](#)
- rdstate, [1903](#)
- read, [1903](#)
- readsome, [1904](#)
- register_callback, [1905](#)
- right, [1918](#)
- scientific, [1918](#)
- seekdir, [1868](#)
- seekg, [1905](#)
- seekp, [1906](#)
- setf, [1907](#)
- setstate, [1908](#)
- showbase, [1919](#)
- showpoint, [1919](#)
- showpos, [1919](#)
- skipws, [1919](#)
- sync, [1908](#)
- sync_with_stdio, [1908](#)
- tellg, [1909](#)
- tellp, [1909](#)
- tie, [1909](#), [1910](#)
- trunc, [1919](#)
- unget, [1910](#)
- unitbuf, [1919](#)
- unsetf, [1911](#)
- uppercase, [1920](#)
- widen, [1911](#)
- width, [1912](#)
- write, [1912](#)
- xalloc, [1913](#)
- std::basic_istream< _CharT, _Traits >, [1860](#)
- std::basic_istream
 - _M_gcount, [1960](#)
 - _M_getloc, [1929](#)
 - __num_put_type, [1926](#)
 - ~basic_istream, [1928](#)
 - adjustfield, [1960](#)
 - app, [1960](#)
 - ate, [1960](#)
 - bad, [1929](#)
 - badbit, [1961](#)
 - basefield, [1961](#)
 - basic_istream, [1928](#)
 - beg, [1961](#)
 - binary, [1961](#)
 - boolalpha, [1961](#)
 - clear, [1929](#)
 - copyfmt, [1929](#)
 - cur, [1962](#)
 - dec, [1962](#)
 - end, [1962](#)
 - eof, [1930](#)
 - eofbit, [1962](#)
 - event, [1928](#)
 - event_callback, [1926](#)
 - exceptions, [1930](#)

- fail, [1931](#)
- failbit, [1962](#)
- fill, [1931](#)
- fixed, [1963](#)
- flags, [1932](#)
- floatfield, [1963](#)
- fmtflags, [1926](#)
- gcount, [1933](#)
- get, [1933–1935](#)
- getline, [1936](#), [1937](#)
- getloc, [1937](#)
- good, [1937](#)
- goodbit, [1963](#)
- hex, [1963](#)
- ignore, [1937](#), [1938](#)
- imbue, [1939](#)
- in, [1964](#)
- init, [1939](#)
- internal, [1964](#)
- iostate, [1927](#)
- isword, [1939](#)
- left, [1964](#)
- narrow, [1940](#)
- oct, [1964](#)
- openmode, [1927](#)
- operator bool, [1940](#)
- operator!, [1941](#)
- operator>>, [1941–1943](#), [1945–1948](#)
- out, [1964](#)
- peek, [1948](#)
- precision, [1948](#), [1949](#)
- putback, [1949](#)
- pword, [1950](#)
- rdbuf, [1950](#)
- rdstate, [1951](#)
- read, [1951](#)
- readsome, [1952](#)
- register_callback, [1953](#)
- right, [1965](#)
- scientific, [1965](#)
- seekdir, [1927](#)
- seekg, [1953](#), [1954](#)
- setf, [1954](#), [1955](#)
- setstate, [1955](#)
- showbase, [1965](#)
- showpoint, [1965](#)
- showpos, [1965](#)
- skipws, [1965](#)
- sync, [1955](#)
- sync_with_stdio, [1956](#)
- tellg, [1956](#)
- tie, [1957](#)
- trunc, [1966](#)
- unget, [1957](#)
- unitbuf, [1966](#)
- unsetf, [1958](#)
- uppercase, [1966](#)
- widen, [1958](#)
- width, [1959](#)
- xalloc, [1960](#)
- std::basic_istream<_CharT, _Traits >, [1920](#)
- std::basic_istream<_CharT, _Traits >::sentry, [1966](#)
- std::basic_istream::sentry
 - operator bool, [1968](#)
 - sentry, [1967](#)
 - traits_type, [1967](#)
- std::basic_istream
 - _M_gcount, [2010](#)
 - _M_getloc, [1978](#)
 - __num_put_type, [1974](#)
 - ~basic_istream, [1977](#)
 - adjustfield, [2010](#)
 - app, [2010](#)
 - ate, [2011](#)
 - bad, [1978](#)
 - badbit, [2011](#)
 - basefield, [2011](#)
 - basic_istream, [1977](#)
 - beg, [2011](#)
 - binary, [2011](#)
 - boolalpha, [2012](#)
 - clear, [1978](#)
 - copyfmt, [1979](#)
 - cur, [2012](#)
 - dec, [2012](#)
 - end, [2012](#)
 - eof, [1979](#)
 - eofbit, [2012](#)
 - event, [1977](#)
 - event_callback, [1974](#)
 - exceptions, [1979](#), [1980](#)
 - fail, [1980](#)
 - failbit, [2013](#)
 - fill, [1981](#)
 - fixed, [2013](#)
 - flags, [1981](#), [1982](#)
 - floatfield, [2013](#)
 - fmtflags, [1975](#)
 - gcount, [1982](#)
 - get, [1982–1985](#)
 - getline, [1986](#), [1987](#)
 - getloc, [1987](#)
 - good, [1987](#)
 - goodbit, [2013](#)
 - hex, [2014](#)
 - ignore, [1987](#), [1988](#)
 - imbue, [1989](#)
 - in, [2014](#)

- init, [1989](#)
- internal, [2014](#)
- iostate, [1975](#)
- isword, [1989](#)
- left, [2014](#)
- narrow, [1990](#)
- oct, [2015](#)
- openmode, [1976](#)
- operator bool, [1990](#)
- operator!, [1991](#)
- operator>>, [1991–1993](#), [1995–1998](#)
- out, [2015](#)
- peek, [1998](#)
- precision, [1998](#), [1999](#)
- putback, [1999](#)
- pword, [2000](#)
- rdbuf, [2000](#), [2001](#)
- rdstate, [2001](#)
- read, [2001](#)
- readsome, [2002](#)
- register_callback, [2003](#)
- right, [2015](#)
- scientific, [2015](#)
- seekdir, [1976](#)
- seekg, [2003](#), [2004](#)
- setf, [2004](#), [2005](#)
- setstate, [2005](#)
- showbase, [2015](#)
- showpoint, [2015](#)
- showpos, [2016](#)
- skipws, [2016](#)
- str, [2005](#), [2006](#)
- sync, [2006](#)
- sync_with_stdio, [2006](#)
- tellg, [2007](#)
- tie, [2007](#)
- trunc, [2016](#)
- unget, [2008](#)
- unitbuf, [2016](#)
- unsetf, [2008](#)
- uppercase, [2016](#)
- widen, [2009](#)
- width, [2009](#)
- xalloc, [2010](#)
- std::basic_istream<_CharT, _Traits, _Alloc>, [1969](#)
- std::basic_ofstream
 - _M_getloc, [2025](#)
 - _M_write, [2025](#)
 - __num_get_type, [2022](#)
 - ~basic_ofstream, [2025](#)
- adjustfield, [2051](#)
- app, [2051](#)
- ate, [2051](#)
- bad, [2026](#)
- badbit, [2051](#)
- basefield, [2052](#)
- basic_ofstream, [2024](#), [2025](#)
- beg, [2052](#)
- binary, [2052](#)
- boolalpha, [2052](#)
- clear, [2026](#)
- close, [2026](#)
- copyfmt, [2027](#)
- cur, [2052](#)
- dec, [2053](#)
- end, [2053](#)
- eof, [2027](#)
- eofbit, [2053](#)
- event, [2024](#)
- event_callback, [2022](#)
- exceptions, [2027](#), [2028](#)
- fail, [2028](#)
- failbit, [2053](#)
- fill, [2029](#)
- fixed, [2053](#)
- flags, [2029](#), [2030](#)
- floatfield, [2054](#)
- flush, [2030](#)
- fmtflags, [2022](#)
- getloc, [2030](#)
- good, [2031](#)
- goodbit, [2054](#)
- hex, [2054](#)
- imbue, [2031](#)
- in, [2054](#)
- init, [2032](#)
- internal, [2054](#)
- iostate, [2023](#)
- is_open, [2032](#)
- isword, [2032](#)
- left, [2055](#)
- narrow, [2032](#)
- oct, [2055](#)
- open, [2033](#)
- openmode, [2023](#)
- operator bool, [2034](#)
- operator!, [2034](#)
- operator<<, [2034–2037](#), [2039–2041](#)
- out, [2055](#)
- precision, [2041](#), [2042](#)
- put, [2042](#)
- pword, [2043](#)
- rdbuf, [2043](#), [2044](#)
- rdstate, [2044](#)
- register_callback, [2044](#)
- right, [2055](#)
- scientific, [2055](#)
- seekdir, [2024](#)

- seekp, 2045
- setf, 2045, 2046
- setstate, 2046
- showbase, 2056
- showpoint, 2056
- showpos, 2056
- skipws, 2056
- sync_with_stdio, 2047
- tellp, 2047
- tie, 2047, 2048
- trunc, 2056
- unitbuf, 2056
- unsetf, 2048
- uppercase, 2057
- widen, 2048
- width, 2049
- write, 2050
- xalloc, 2050
- std::basic_ofstream< _CharT, _Traits >, 2017
- std::basic_ostream
 - _M_getloc, 2065
 - _M_write, 2065
 - __num_get_type, 2062
 - ~basic_ostream, 2065
 - adjustfield, 2088
 - app, 2088
 - ate, 2088
 - bad, 2066
 - badbit, 2089
 - basefield, 2089
 - basic_ostream, 2065
 - beg, 2089
 - binary, 2089
 - boolalpha, 2089
 - clear, 2066
 - copyfmt, 2066
 - cur, 2090
 - dec, 2090
 - end, 2090
 - eof, 2067
 - eofbit, 2090
 - event, 2064
 - event_callback, 2062
 - exceptions, 2067
 - fail, 2068
 - failbit, 2090
 - fill, 2068
 - fixed, 2091
 - flags, 2069
 - floatfield, 2091
 - flush, 2070
 - fmtflags, 2063
 - getloc, 2070
 - good, 2070
 - goodbit, 2091
 - hex, 2091
 - imbue, 2070
 - in, 2092
 - init, 2071
 - internal, 2092
 - iostate, 2063
 - isword, 2071
 - left, 2092
 - narrow, 2071
 - oct, 2092
 - openmode, 2063
 - operator bool, 2072
 - operator!, 2072
 - operator<<, 2072–2078
 - out, 2092
 - precision, 2079
 - put, 2079
 - pword, 2080
 - rdbuf, 2080, 2081
 - rdstate, 2081
 - register_callback, 2082
 - right, 2093
 - scientific, 2093
 - seekdir, 2064
 - seekp, 2082
 - setf, 2083
 - setstate, 2084
 - showbase, 2093
 - showpoint, 2093
 - showpos, 2093
 - skipws, 2093
 - sync_with_stdio, 2084
 - tellp, 2084
 - tie, 2085
 - trunc, 2094
 - unitbuf, 2094
 - unsetf, 2086
 - uppercase, 2094
 - widen, 2086
 - width, 2086, 2087
 - write, 2087
 - xalloc, 2088
- std::basic_ostream< _CharT, _Traits >, 2057
- std::basic_ostream< _CharT, _Traits >::sentry, 2094
- std::basic_ostream::sentry
 - ~sentry, 2095
 - operator bool, 2096
 - sentry, 2095
- std::basic_ostringstream
 - _M_getloc, 2105
 - _M_write, 2105
 - __num_get_type, 2101
 - ~basic_ostringstream, 2104

- adjustfield, [2128](#)
- app, [2128](#)
- ate, [2129](#)
- bad, [2105](#)
- badbit, [2129](#)
- basefield, [2129](#)
- basic_ostringstream, [2104](#)
- beg, [2129](#)
- binary, [2129](#)
- boolalpha, [2130](#)
- clear, [2106](#)
- copyfmt, [2106](#)
- cur, [2130](#)
- dec, [2130](#)
- end, [2130](#)
- eof, [2106](#)
- eofbit, [2130](#)
- event, [2104](#)
- event_callback, [2101](#)
- exceptions, [2107](#)
- fail, [2108](#)
- failbit, [2131](#)
- fill, [2108](#)
- fixed, [2131](#)
- flags, [2109](#)
- floatfield, [2131](#)
- flush, [2109](#)
- fmtflags, [2102](#)
- getloc, [2110](#)
- good, [2110](#)
- goodbit, [2131](#)
- hex, [2132](#)
- imbue, [2110](#)
- in, [2132](#)
- init, [2111](#)
- internal, [2132](#)
- iostate, [2102](#)
- isword, [2111](#)
- left, [2132](#)
- narrow, [2111](#)
- oct, [2133](#)
- openmode, [2103](#)
- operator bool, [2112](#)
- operator!, [2112](#)
- operator<<, [2112–2118](#)
- out, [2133](#)
- precision, [2119](#)
- put, [2119](#)
- pword, [2120](#)
- rdbuf, [2120, 2121](#)
- rdstate, [2121](#)
- register_callback, [2121](#)
- right, [2133](#)
- scientific, [2133](#)
- seekdir, [2103](#)
- seekp, [2122](#)
- setf, [2123](#)
- setstate, [2123](#)
- showbase, [2133](#)
- showpoint, [2133](#)
- showpos, [2134](#)
- skipws, [2134](#)
- str, [2124](#)
- sync_with_stdio, [2124](#)
- tellp, [2125](#)
- tie, [2125](#)
- trunc, [2134](#)
- unitbuf, [2134](#)
- unsetf, [2126](#)
- uppercase, [2134](#)
- widen, [2126](#)
- width, [2127](#)
- write, [2127](#)
- xalloc, [2128](#)
- std::basic_ostringstream<_CharT, _Traits, _Alloc >, [2096](#)
- std::basic_regex
 - ~basic_regex, [2139](#)
 - assign, [2139–2141](#)
 - basic_regex, [2136–2138](#)
 - flags, [2142](#)
 - getloc, [2142](#)
 - imbue, [2142](#)
 - mark_count, [2142](#)
 - operator=, [2142, 2143](#)
 - swap, [2144](#)
- std::basic_regex<_Ch_type, _Rx_traits >, [2135](#)
- std::basic_streambuf
 - _M_buf_locale, [2164](#)
 - _M_in_beg, [2164](#)
 - _M_in_cur, [2164](#)
 - _M_in_end, [2164](#)
 - _M_out_beg, [2164](#)
 - _M_out_cur, [2164](#)
 - _M_out_end, [2165](#)
 - __streambuf_type, [2148](#)
 - ~basic_streambuf, [2149](#)
 - basic_streambuf, [2149](#)
 - char_type, [2148](#)
 - eback, [2149](#)
 - egptr, [2149](#)
 - epptr, [2150](#)
 - gbump, [2150](#)
 - getloc, [2151](#)
 - gptr, [2151](#)
 - imbue, [2151](#)
 - in_avail, [2152](#)
 - int_type, [2148](#)
 - off_type, [2148](#)

- overflow, 2152
- pbackfail, 2152
- pbase, 2153
- pbump, 2153
- pos_type, 2148
- pptr, 2154
- pubimbue, 2154
- pubseekoff, 2154
- pubseekpos, 2155
- pubsetbuf, 2155
- pubsync, 2155
- sbumpc, 2155
- seekoff, 2156
- seekpos, 2156
- setbuf, 2156
- setg, 2157
- setp, 2157
- sgetc, 2158
- sgetn, 2158
- showmanyc, 2158
- snextc, 2159
- sputbackc, 2159
- sputc, 2160
- sputn, 2160
- sungetc, 2161
- sync, 2161
- traits_type, 2148
- uflow, 2161
- underflow, 2162
- xsgetn, 2162
- xspn, 2163
- std::basic_streambuf< _CharT, _Traits >, 2144
- std::basic_string
 - ~basic_string, 2173
 - append, 2173–2176
 - assign, 2176–2179
 - at, 2180
 - back, 2181
 - basic_string, 2170–2173
 - begin, 2181
 - c_str, 2181
 - capacity, 2182
 - cbegin, 2182
 - cend, 2182
 - clear, 2182
 - compare, 2182–2185
 - copy, 2185
 - crbegin, 2186
 - crend, 2186
 - data, 2186
 - empty, 2187
 - end, 2187
 - erase, 2187, 2188
 - find, 2189, 2190
 - find_first_not_of, 2191, 2192
 - find_first_of, 2192–2194
 - find_last_not_of, 2194–2196
 - find_last_of, 2196, 2197
 - front, 2198
 - get_allocator, 2198
 - insert, 2198–2202
 - length, 2203
 - max_size, 2203
 - npos, 2219
 - operator+=, 2203, 2204
 - operator=, 2205, 2206
 - operator[], 2206, 2207
 - pop_back, 2207
 - push_back, 2207
 - rbegin, 2207, 2208
 - rend, 2208
 - replace, 2208–2214
 - reserve, 2215
 - resize, 2215, 2216
 - rfind, 2216, 2217
 - shrink_to_fit, 2218
 - size, 2218
 - substr, 2218
 - swap, 2219
- std::basic_string< _CharT, _Traits, _Alloc >, 2165
- std::basic_stringbuf
 - _M_buf_locale, 2238
 - _M_in_beg, 2238
 - _M_in_cur, 2239
 - _M_in_end, 2239
 - _M_mode, 2239
 - _M_out_beg, 2239
 - _M_out_cur, 2239
 - _M_out_end, 2239
 - basic_stringbuf, 2223
 - eback, 2223
 - egptr, 2223
 - eptr, 2224
 - gbump, 2224
 - getloc, 2225
 - gptr, 2225
 - imbue, 2225
 - in_avail, 2226
 - overflow, 2226
 - pbackfail, 2227
 - pbase, 2227
 - pbump, 2228
 - pptr, 2228
 - pubimbue, 2228
 - pubseekoff, 2229
 - pubseekpos, 2229
 - pubsetbuf, 2229
 - pubsync, 2230

- sbumpc, [2230](#)
- seekoff, [2230](#)
- seekpos, [2230](#)
- setbuf, [2231](#)
- setg, [2231](#)
- setp, [2232](#)
- sgetc, [2232](#)
- sgetn, [2232](#)
- showmanyc, [2233](#)
- snextc, [2233](#)
- sputbackc, [2233](#)
- sputc, [2234](#)
- sputn, [2234](#)
- str, [2235](#)
- sungetc, [2235](#)
- sync, [2236](#)
- uflow, [2236](#)
- underflow, [2236](#)
- xsgetn, [2237](#)
- xspn, [2238](#)
- std::basic_stringbuf<_CharT, _Traits, _Alloc >, [2220](#)
- std::basic_stringstream
 - _M_gcount, [2293](#)
 - _M_getloc, [2251](#)
 - _M_write, [2251](#)
 - __num_put_type, [2247](#)
 - ~basic_stringstream, [2251](#)
 - adjustfield, [2293](#)
 - app, [2293](#)
 - ate, [2293](#)
 - bad, [2252](#)
 - badbit, [2294](#)
 - basefield, [2294](#)
 - basic_stringstream, [2249](#)
 - beg, [2294](#)
 - binary, [2294](#)
 - boolalpha, [2294](#)
 - clear, [2252](#)
 - copyfmt, [2252](#)
 - cur, [2295](#)
 - dec, [2295](#)
 - end, [2295](#)
 - eof, [2253](#)
 - eofbit, [2295](#)
 - event, [2249](#)
 - event_callback, [2247](#)
 - exceptions, [2253](#)
 - fail, [2254](#)
 - failbit, [2295](#)
 - fill, [2254](#)
 - fixed, [2296](#)
 - flags, [2255](#)
 - floatfield, [2296](#)
 - flush, [2256](#)
 - fmtflags, [2247](#)
 - gcount, [2256](#)
 - get, [2256–2259](#)
 - getline, [2259](#), [2260](#)
 - getloc, [2260](#)
 - good, [2261](#)
 - goodbit, [2296](#)
 - hex, [2296](#)
 - ignore, [2261](#), [2262](#)
 - imbue, [2262](#)
 - in, [2297](#)
 - init, [2263](#)
 - internal, [2297](#)
 - iostate, [2248](#)
 - isw, [2263](#)
 - left, [2297](#)
 - narrow, [2263](#)
 - oct, [2297](#)
 - openmode, [2248](#)
 - operator bool, [2264](#)
 - operator!, [2264](#)
 - operator<<, [2264–2270](#)
 - operator>>, [2271–2273](#), [2275–2278](#)
 - out, [2297](#)
 - peek, [2278](#)
 - precision, [2278](#), [2279](#)
 - put, [2279](#)
 - putback, [2280](#)
 - pwd, [2280](#)
 - rdbuf, [2281](#)
 - rdstate, [2281](#)
 - read, [2282](#)
 - reads, [2282](#)
 - register_callback, [2283](#)
 - right, [2298](#)
 - scientific, [2298](#)
 - seekdir, [2248](#)
 - seekg, [2283](#), [2284](#)
 - seekp, [2285](#)
 - setf, [2286](#)
 - setstate, [2286](#)
 - showbase, [2298](#)
 - showpoint, [2298](#)
 - showpos, [2298](#)
 - skipws, [2298](#)
 - str, [2287](#)
 - sync, [2287](#)
 - sync_with_stdio, [2288](#)
 - tellg, [2288](#)
 - tellp, [2289](#)
 - tie, [2289](#)
 - trunc, [2299](#)
 - unget, [2290](#)
 - unitbuf, [2299](#)

- unsetf, 2290
- uppercase, 2299
- widen, 2291
- width, 2291
- write, 2292
- xalloc, 2292
- std::basic_stringstream< _CharT, _Traits, _Alloc >, 2240
- std::bernoulli_distribution, 2299
 - bernoulli_distribution, 2300
 - max, 2301
 - min, 2301
 - operator(), 2301
 - operator==, 2302
 - p, 2301
 - param, 2301
 - reset, 2302
 - result_type, 2300
- std::bernoulli_distribution::param_type, 2302
- std::bidirectional_iterator_tag, 2303
- std::binary_function
 - first_argument_type, 2304
 - result_type, 2304
 - second_argument_type, 2305
- std::binary_function< _Arg1, _Arg2, _Result >, 2304
- std::binary_negate
 - first_argument_type, 2306
 - result_type, 2306
 - second_argument_type, 2306
- std::binary_negate< _Predicate >, 2305
- std::binder1st
 - argument_type, 2308
 - result_type, 2308
- std::binder1st< _Operation >, 2307
- std::binder2nd
 - argument_type, 2309
 - result_type, 2309
- std::binder2nd< _Operation >, 2308
- std::binomial_distribution
 - max, 2311
 - min, 2311
 - operator<<, 2312
 - operator>>, 2313
 - operator(), 2311
 - operator==, 2313
 - p, 2311
 - param, 2312
 - reset, 2312
 - result_type, 2311
 - t, 2312
- std::binomial_distribution< _IntType >, 2309
- std::binomial_distribution< _IntType >::param_type, 2313
- std::bitset
 - all, 2319
 - any, 2319
 - bitset, 2318, 2319
 - count, 2320
 - flip, 2320
 - none, 2320
 - operator!=, 2320
 - operator<<, 2321
 - operator<=, 2321
 - operator>>, 2321
 - operator>=, 2322
 - operator^=, 2323
 - operator==, 2321
 - operator&=, 2321
 - operator[], 2322
 - operator|=, 2323
 - operator~, 2323
 - reset, 2323, 2324
 - set, 2324
 - size, 2324
 - test, 2325
 - to_string, 2325
 - to_ulong, 2325
- std::bitset< _Nb >, 2314
- std::bitset< _Nb >::reference, 2326
- std::cauchy_distribution
 - max, 2328
 - min, 2328
 - operator(), 2328
 - operator==, 2329
 - param, 2328
 - reset, 2329
 - result_type, 2328
- std::cauchy_distribution< _RealType >, 2327
- std::cauchy_distribution< _RealType >::param_type, 2329
- std::char_traits< __gnu_cxx::character< _Value, _Int, _St > >, 2331
- std::char_traits< _CharT >, 2330
- std::char_traits< char >, 2332
- std::char_traits< wchar_t >, 2333
- std::chi_squared_distribution
 - max, 2336
 - min, 2336
 - operator<<, 2337
 - operator>>, 2337
 - operator(), 2336
 - operator==, 2337
 - param, 2336
 - reset, 2337
 - result_type, 2336
- std::chi_squared_distribution< _RealType >, 2334
- std::chi_squared_distribution< _RealType >::param_type, 2338
- std::chrono, 679
 - duration_cast, 682

- hours, [681](#)
- microseconds, [681](#)
- milliseconds, [682](#)
- minutes, [682](#)
- nanoseconds, [682](#)
- seconds, [682](#)
- time_point_cast, [682](#)
- std::chrono::_V2::steady_clock, [2339](#)
- std::chrono::_V2::system_clock, [2339](#)
- std::chrono::duration< _Rep, _Period >, [2340](#)
- std::chrono::duration_values< _Rep >, [2341](#)
- std::chrono::time_point< _Clock, _Dur >, [2341](#)
- std::chrono::treat_as_floating_point< _Rep >, [2342](#)
- std::codecvt
 - do_out, [2345](#)
 - in, [2345](#)
 - out, [2346](#)
 - unshift, [2347](#)
- std::codecvt< _InternT, _ExternT, _StateT >, [2343](#)
- std::codecvt< _InternT, _ExternT, encoding_state >, [2348](#)
 - do_out, [2349](#)
 - in, [2349](#)
 - out, [2350](#)
 - unshift, [2351](#)
- std::codecvt< char, char, mbstate_t >, [2352](#)
 - do_out, [2354](#)
 - in, [2354](#)
 - out, [2355](#)
 - unshift, [2355](#)
- std::codecvt< char16_t, char, mbstate_t >, [2357](#)
 - do_out, [2358](#)
 - in, [2358](#)
 - out, [2359](#)
 - unshift, [2360](#)
- std::codecvt< char32_t, char, mbstate_t >, [2361](#)
 - do_out, [2362](#)
 - in, [2362](#)
 - out, [2363](#)
 - unshift, [2364](#)
- std::codecvt< wchar_t, char, mbstate_t >, [2365](#)
 - do_out, [2367](#)
 - in, [2367](#)
 - out, [2368](#)
 - unshift, [2368](#)
- std::codecvt_base, [2370](#)
- std::codecvt_byname
 - do_out, [2373](#)
 - in, [2373](#)
 - out, [2374](#)
 - unshift, [2374](#)
- std::codecvt_byname< _InternT, _ExternT, _StateT >, [2371](#)
- std::collate
 - ~collate, [2378](#)
- char_type, [2377](#)
- collate, [2377](#)
- compare, [2378](#)
- do_compare, [2378](#)
- do_hash, [2379](#)
- do_transform, [2379](#)
- hash, [2380](#)
- id, [2381](#)
- string_type, [2377](#)
- transform, [2380](#)
- std::collate< _CharT >, [2375](#)
- std::collate_byname
 - char_type, [2383](#)
 - compare, [2383](#)
 - do_compare, [2383](#)
 - do_hash, [2384](#)
 - do_transform, [2385](#)
 - hash, [2385](#)
 - id, [2386](#)
 - string_type, [2383](#)
 - transform, [2385](#)
- std::collate_byname< _CharT >, [2381](#)
- std::complex
 - complex, [2388](#)
 - operator+=, [2388](#)
 - operator-=, [2388](#)
 - value_type, [2387](#)
- std::complex< _Tp >, [2386](#)
- std::complex< double >, [2388](#)
- std::complex< float >, [2389](#)
- std::complex< long double >, [2390](#)
- std::condition_variable, [2391](#)
- std::const_mem_fun1_ref_t
 - first_argument_type, [2393](#)
 - result_type, [2393](#)
 - second_argument_type, [2393](#)
- std::const_mem_fun1_ref_t< _Ret, _Tp, _Arg >, [2392](#)
- std::const_mem_fun1_t
 - first_argument_type, [2394](#)
 - result_type, [2394](#)
 - second_argument_type, [2395](#)
- std::const_mem_fun1_t< _Ret, _Tp, _Arg >, [2394](#)
- std::const_mem_fun_ref_t
 - argument_type, [2396](#)
 - result_type, [2396](#)
- std::const_mem_fun_ref_t< _Ret, _Tp >, [2395](#)
- std::const_mem_fun_t
 - argument_type, [2397](#)
 - result_type, [2397](#)
- std::const_mem_fun_t< _Ret, _Tp >, [2396](#)
- std::ctype
 - do_is, [2400](#)
 - do_narrow, [2401](#)
 - do_scan_is, [2402](#)

- do_scan_not, [2402](#)
- do_tolower, [2403](#)
- do_toupper, [2404](#)
- do_widen, [2405](#)
- id, [2412](#)
- is, [2406](#)
- narrow, [2407](#)
- scan_is, [2408](#)
- scan_not, [2408](#)
- tolower, [2409](#)
- toupper, [2410](#)
- widen, [2411](#)
- std::ctype< _CharT >, [2398](#)
- std::ctype< char >, [2412](#)
 - ~ctype, [2415](#)
 - char_type, [2414](#)
 - classic_table, [2415](#)
 - ctype, [2414](#), [2415](#)
 - do_narrow, [2415](#), [2416](#)
 - do_tolower, [2416](#), [2417](#)
 - do_toupper, [2417](#), [2418](#)
 - do_widen, [2418](#)
 - id, [2425](#)
 - is, [2419](#)
 - narrow, [2420](#)
 - scan_is, [2421](#)
 - scan_not, [2422](#)
 - table, [2422](#)
 - table_size, [2425](#)
 - tolower, [2422](#), [2423](#)
 - toupper, [2423](#), [2424](#)
 - widen, [2424](#), [2425](#)
- std::ctype< wchar_t >, [2426](#)
 - ~ctype, [2429](#)
 - char_type, [2428](#)
 - ctype, [2428](#), [2429](#)
 - do_is, [2429](#)
 - do_narrow, [2430](#)
 - do_scan_is, [2431](#)
 - do_scan_not, [2432](#)
 - do_tolower, [2432](#), [2433](#)
 - do_toupper, [2433](#)
 - do_widen, [2434](#)
 - id, [2441](#)
 - is, [2435](#)
 - narrow, [2436](#)
 - scan_is, [2437](#)
 - scan_not, [2438](#)
 - tolower, [2438](#), [2439](#)
 - toupper, [2439](#), [2440](#)
 - widen, [2440](#)
- std::ctype_base, [2441](#)
- std::ctype_byname
 - do_is, [2445](#)
 - do_narrow, [2445](#), [2446](#)
 - do_scan_is, [2447](#)
 - do_scan_not, [2447](#)
 - do_tolower, [2448](#)
 - do_toupper, [2449](#)
 - do_widen, [2450](#)
 - id, [2456](#)
 - is, [2451](#)
 - narrow, [2452](#)
 - scan_is, [2452](#)
 - scan_not, [2453](#)
 - tolower, [2453](#), [2454](#)
 - toupper, [2454](#), [2455](#)
 - widen, [2455](#), [2456](#)
- std::ctype_byname< _CharT >, [2443](#)
- std::ctype_byname< char >, [2457](#)
 - char_type, [2459](#)
 - classic_table, [2459](#)
 - do_narrow, [2459](#), [2460](#)
 - do_tolower, [2460](#), [2461](#)
 - do_toupper, [2461](#), [2462](#)
 - do_widen, [2462](#)
 - id, [2469](#)
 - is, [2463](#)
 - narrow, [2464](#)
 - scan_is, [2465](#)
 - scan_not, [2466](#)
 - table, [2466](#)
 - table_size, [2469](#)
 - tolower, [2466](#), [2467](#)
 - toupper, [2467](#), [2468](#)
 - widen, [2468](#), [2469](#)
- std::decimal, [683](#)
 - decimal32_to_long_long, [692](#)
- std::decimal::decimal128, [2470](#)
 - decimal128, [2471](#)
- std::decimal::decimal32, [2472](#)
 - decimal32, [2473](#)
- std::decimal::decimal64, [2473](#)
 - decimal64, [2475](#)
- std::default_delete
 - default_delete, [2476](#)
 - operator(), [2476](#)
- std::default_delete< _Tp >, [2475](#)
- std::default_delete< _Tp[] >, [2476](#)
 - default_delete, [2477](#)
 - operator(), [2477](#)
- std::defer_lock_t, [2477](#)
- std::deque
 - _M_fill_initialize, [2486](#)
 - _M_initialize_map, [2486](#)
 - _M_new_elements_at_back, [2487](#)
 - _M_new_elements_at_front, [2487](#)
 - _M_pop_back_aux, [2487](#)

- `_M_pop_front_aux`, 2487
- `_M_push_back_aux`, 2488
- `_M_push_front_aux`, 2488
- `_M_range_check`, 2488
- `_M_range_initialize`, 2488, 2489
- `_M_reallocate_map`, 2489
- `_M_reserve_elements_at_back`, 2489
- `_M_reserve_elements_at_front`, 2489
- `_M_reserve_map_at_back`, 2490
- `_M_reserve_map_at_front`, 2490
- `~deque`, 2486
- `assign`, 2490, 2491
- `at`, 2491, 2492
- `back`, 2492
- `begin`, 2492, 2493
- `cbegin`, 2493
- `cend`, 2493
- `clear`, 2493
- `crbegin`, 2493
- `crend`, 2493
- `deque`, 2483–2485
- `emplace`, 2493
- `empty`, 2494
- `end`, 2494
- `erase`, 2494, 2495
- `front`, 2495
- `get_allocator`, 2495
- `insert`, 2496–2498
- `max_size`, 2498
- `operator=`, 2498, 2499
- `operator[]`, 2500
- `pop_back`, 2500
- `pop_front`, 2500
- `push_back`, 2501
- `push_front`, 2501
- `rbegin`, 2501
- `rend`, 2502
- `resize`, 2502
- `shrink_to_fit`, 2503
- `size`, 2503
- `swap`, 2503
- `std::deque< _Tp, _Alloc >`, 2478
- `std::discard_block_engine`
 - `base`, 2506
 - `discard`, 2506
 - `discard_block_engine`, 2505, 2506
 - `max`, 2506
 - `min`, 2507
 - `operator<<`, 2508
 - `operator>>`, 2508
 - `operator()`, 2507
 - `operator==`, 2508
 - `result_type`, 2505
 - `seed`, 2507
 - `std::discard_block_engine< _RandomNumberEngine, _P, _R >`, 2503
- `std::discrete_distribution`
 - `max`, 2511
 - `min`, 2511
 - `operator<<`, 2512
 - `operator>>`, 2512
 - `operator()`, 2511
 - `operator==`, 2512
 - `param`, 2511
 - `probabilities`, 2511
 - `reset`, 2512
 - `result_type`, 2510
- `std::discrete_distribution< _IntType >`, 2509
- `std::discrete_distribution< _IntType >::param_type`, 2513
- `std::divides`
 - `first_argument_type`, 2514
 - `result_type`, 2514
 - `second_argument_type`, 2515
- `std::divides< _Tp >`, 2514
- `std::divides< void >`, 2515
- `std::domain_error`, 2516
 - `what`, 2516
- `std::enable_shared_from_this< _Tp >`, 2517
- `std::equal_to`
 - `first_argument_type`, 2518
 - `result_type`, 2518
 - `second_argument_type`, 2518
- `std::equal_to< _Tp >`, 2517
- `std::equal_to< void >`, 2519
- `std::error_code`, 2519
- `std::error_condition`, 2520
- `std::exception`, 2521
 - `what`, 2522
- `std::experimental::filesystem::v1::path`, 2522
- `std::experimental::filesystem::v1::path::iterator`, 2524
- `std::experimental::fundamentals_v1::_Has_addressof< _Tp >`, 2525
- `std::experimental::fundamentals_v1::_Optional_base< _Tp, _ShouldProvideDestructor >`, 2526
- `std::experimental::fundamentals_v1::_Optional_base< _Tp, false >`, 2527
- `std::experimental::fundamentals_v1::any`, 2528
 - `~any`, 2529
 - `any`, 2528, 2529
 - `clear`, 2529
 - `empty`, 2529
 - `operator=`, 2530
 - `swap`, 2530
 - `type`, 2530
- `std::experimental::fundamentals_v1::bad_any_cast`, 2531
 - `what`, 2531
- `std::experimental::fundamentals_v1::bad_optional_`
 - `access`, 2532

- what, [2532](#)
- std::experimental::fundamentals_v1::basic_string_view< _CharT, _Traits >, [2533](#)
- std::experimental::fundamentals_v1::in_place_t, [2535](#)
- std::experimental::fundamentals_v1::nullopt_t, [2535](#)
- std::experimental::fundamentals_v1::optional< _Tp >, [2536](#)
- std::experimental::fundamentals_v2::_Not_fn< _Fn >, [2538](#)
- std::experimental::fundamentals_v2::ostream_joiner< _< DelimT, _CharT, _Traits >, [2539](#)
- std::experimental::fundamentals_v2::owner_less< shared< _ptr< _Tp > >, [2539](#)
 - first_argument_type, [2540](#)
 - result_type, [2540](#)
 - second_argument_type, [2540](#)
- std::experimental::fundamentals_v2::owner_less< weak< _ptr< _Tp > >, [2541](#)
 - first_argument_type, [2541](#)
 - result_type, [2541](#)
 - second_argument_type, [2541](#)
- std::experimental::fundamentals_v2::propagate_const< _Tp >, [2542](#)
- std::exponential_distribution
 - exponential_distribution, [2544](#)
 - lambda, [2545](#)
 - max, [2545](#)
 - min, [2545](#)
 - operator(), [2545](#)
 - operator==, [2546](#)
 - param, [2545](#)
 - reset, [2546](#)
 - result_type, [2544](#)
- std::exponential_distribution< _RealType >, [2543](#)
- std::exponential_distribution< _RealType >::param_type, [2546](#)
- std::extreme_value_distribution
 - a, [2548](#)
 - b, [2548](#)
 - max, [2548](#)
 - min, [2549](#)
 - operator(), [2549](#)
 - operator==, [2550](#)
 - param, [2549](#)
 - reset, [2549](#)
 - result_type, [2548](#)
- std::extreme_value_distribution< _RealType >, [2547](#)
- std::extreme_value_distribution< _RealType >::param_type, [2550](#)
- std::fisher_f_distribution
 - max, [2552](#)
 - min, [2552](#)
 - operator<=, [2553](#)
 - operator>, [2553](#)
 - operator(), [2552](#)
 - operator==, [2553](#)
 - param, [2552](#)
 - reset, [2553](#)
 - result_type, [2552](#)
- std::fisher_f_distribution< _RealType >, [2551](#)
- std::fisher_f_distribution< _RealType >::param_type, [2554](#)
- std::forward_iterator_tag, [2555](#)
- std::forward_list
 - ~forward_list, [2561](#)
 - assign, [2562](#)
 - before_begin, [2563](#)
 - begin, [2563](#)
 - cbefore_begin, [2563](#)
 - cbegin, [2563](#)
 - cend, [2564](#)
 - clear, [2564](#)
 - emplace_after, [2564](#)
 - emplace_front, [2564](#)
 - empty, [2565](#)
 - end, [2565](#)
 - erase_after, [2565](#), [2566](#)
 - forward_list, [2559](#)–[2561](#)
 - front, [2566](#)
 - get_allocator, [2566](#)
 - insert_after, [2567](#), [2568](#)
 - max_size, [2568](#)
 - merge, [2569](#)
 - operator=, [2569](#), [2570](#)
 - pop_front, [2570](#)
 - push_front, [2570](#)
 - remove, [2571](#)
 - remove_if, [2571](#)
 - resize, [2571](#), [2572](#)
 - reverse, [2572](#)
 - sort, [2572](#)
 - splice_after, [2573](#), [2575](#)
 - swap, [2575](#)
 - unique, [2575](#), [2576](#)
- std::forward_list< _Tp, _Alloc >, [2556](#)
- std::fpos
 - fpos, [2577](#)
 - operator streamoff, [2577](#)
 - operator+, [2577](#)
 - operator+=, [2577](#)
 - operator-, [2577](#), [2578](#)
 - operator-=, [2578](#)
 - state, [2578](#)
- std::fpos< _StateT >, [2576](#)
- std::front_insert_iterator
 - container_type, [2579](#)
 - difference_type, [2579](#)
 - front_insert_iterator, [2580](#)

- iterator_category, 2580
- operator*, 2580
- operator++, 2580, 2581
- operator=, 2581
- pointer, 2580
- reference, 2580
- value_type, 2580
- std::front_insert_iterator< _Container >, 2578
- std::function< _Res(_ArgTypes...)>, 2582
 - function, 2583, 2584
 - operator bool, 2585
 - operator(), 2585
 - operator=, 2585–2587
 - swap, 2587
 - target, 2587, 2588
 - target_type, 2588
- std::future
 - _M_get_result, 2590
 - _Ptr, 2590
 - future, 2590
 - get, 2590
- std::future< _Res >, 2588
- std::future< _Res & >, 2591
 - _M_get_result, 2593
 - _Ptr, 2592
 - future, 2593
 - get, 2593
- std::future< void >, 2593
 - _M_get_result, 2595
 - _Ptr, 2595
 - future, 2595
 - get, 2595
- std::future_error, 2596
 - what, 2596
- std::gamma_distribution
 - alpha, 2598
 - beta, 2598
 - gamma_distribution, 2598
 - max, 2598
 - min, 2599
 - operator<<, 2600
 - operator>>, 2600
 - operator(), 2599
 - operator==, 2600
 - param, 2599
 - reset, 2600
 - result_type, 2598
- std::gamma_distribution< _RealType >, 2597
- std::gamma_distribution< _RealType >::param_type, 2601
- std::geometric_distribution
 - max, 2603
 - min, 2603
 - operator(), 2603
 - operator==, 2604
 - p, 2603
 - param, 2603
 - reset, 2604
 - result_type, 2603
- std::geometric_distribution< _IntType >, 2602
- std::geometric_distribution< _IntType >::param_type, 2604
- std::greater
 - first_argument_type, 2606
 - result_type, 2606
 - second_argument_type, 2606
- std::greater< _Tp >, 2605
- std::greater< void >, 2606
- std::greater_equal
 - first_argument_type, 2607
 - result_type, 2607
 - second_argument_type, 2608
- std::greater_equal< _Tp >, 2607
- std::greater_equal< void >, 2608
- std::gslice, 2609
- std::gslice_array< _Tp >, 2609
- std::hash< __debug::bitset< _Nb > >, 2611
- std::hash< __debug::vector< bool, _Alloc > >, 2612
- std::hash< __gnu_cxx::_u16vstring >, 2612
- std::hash< __gnu_cxx::_u32vstring >, 2613
- std::hash< __gnu_cxx::_vstring >, 2614
- std::hash< __gnu_cxx::_wvstring >, 2614
- std::hash< __gnu_cxx::throw_value_limit >, 2615
 - argument_type, 2616
 - result_type, 2616
- std::hash< __gnu_cxx::throw_value_random >, 2616
 - argument_type, 2617
 - result_type, 2617
- std::hash< __profile::bitset< _Nb > >, 2617
- std::hash< __profile::vector< bool, _Alloc > >, 2618
- std::hash< __shared_ptr< _Tp, _Lp > >, 2619
- std::hash< _Tp >, 2611
- std::hash< _Tp * >, 2619
- std::hash< bool >, 2620
- std::hash< char >, 2620
- std::hash< char16_t >, 2621
- std::hash< char32_t >, 2622
- std::hash< double >, 2622
- std::hash< error_code >, 2623
- std::hash< experimental::shared_ptr< _Tp > >, 2623
- std::hash< float >, 2624
- std::hash< int >, 2625
- std::hash< long >, 2625
- std::hash< long double >, 2626
- std::hash< long long >, 2626
- std::hash< shared_ptr< _Tp > >, 2627
- std::hash< short >, 2628
- std::hash< signed char >, 2628

- std::hash< string >, 2629
- std::hash< thread::id >, 2629
- std::hash< type_index >, 2630
- std::hash< u16string >, 2631
- std::hash< u32string >, 2631
- std::hash< unique_ptr< _Tp, _Dp > >, 2632
- std::hash< unsigned char >, 2632
- std::hash< unsigned int >, 2633
- std::hash< unsigned long >, 2634
- std::hash< unsigned long long >, 2634
- std::hash< unsigned short >, 2635
- std::hash< wchar_t >, 2635
- std::hash< wstring >, 2636
- std::hash<::bitset< _Nb > >, 2637
- std::hash<::vector< bool, _Alloc > >, 2637
- std::independent_bits_engine
 - base, 2640
 - discard, 2641
 - independent_bits_engine, 2639, 2640
 - max, 2641
 - min, 2641
 - operator>>, 2642
 - operator(), 2641
 - operator==, 2642
 - result_type, 2639
 - seed, 2641, 2642
- std::independent_bits_engine< _RandomNumberEngine, __w, _UIntType >, 2638
- std::indirect_array< _Tp >, 2643
- std::initializer_list< _E >, 2645
- std::input_iterator_tag, 2646
- std::insert_iterator
 - container_type, 2648
 - difference_type, 2648
 - insert_iterator, 2649
 - iterator_category, 2648
 - operator*, 2649
 - operator++, 2649
 - operator=, 2649
 - pointer, 2648
 - reference, 2648
 - value_type, 2648
- std::insert_iterator< _Container >, 2647
- std::integer_sequence< _Tp, _Idx >, 2650
- std::integral_constant< _Tp, __v >, 2651
- std::invalid_argument, 2652
 - what, 2653
- std::ios_base, 2653
 - _M_getloc, 2659
 - ~ios_base, 2659
 - adjustfield, 2665
 - app, 2665
 - ate, 2665
 - badbit, 2665
 - basefield, 2665
 - beg, 2665
 - binary, 2666
 - boolalpha, 2666
 - cur, 2666
 - dec, 2666
 - end, 2666
 - eofbit, 2667
 - event, 2658
 - event_callback, 2656
 - failbit, 2667
 - fixed, 2667
 - flags, 2659
 - floatfield, 2667
 - fmtflags, 2656
 - getloc, 2660
 - goodbit, 2668
 - hex, 2668
 - imbue, 2660
 - in, 2668
 - internal, 2668
 - iostate, 2657
 - isword, 2660
 - left, 2669
 - oct, 2669
 - openmode, 2657
 - out, 2669
 - precision, 2661
 - pword, 2661
 - register_callback, 2662
 - right, 2669
 - scientific, 2669
 - seekdir, 2658
 - setf, 2662, 2663
 - showbase, 2670
 - showpoint, 2670
 - showpos, 2670
 - skipws, 2670
 - sync_with_stdio, 2663
 - trunc, 2670
 - unitbuf, 2670
 - unsetf, 2663
 - uppercase, 2671
 - width, 2664
 - xalloc, 2664
- std::ios_base::failure, 2671
 - what, 2672
- std::is_abstract< _Tp >, 2672
- std::is_arithmetic< _Tp >, 2673
- std::is_array< typename >, 2674
- std::is_bind_expression< _Bind< _Signature > >, 2676
- std::is_bind_expression< _Bind_result< _Result, _Signature > >, 2677
- std::is_bind_expression< _Tp >, 2675

- `std::is_bind_expression< const _Bind< _Signature > >`, 2678
- `std::is_bind_expression< const _Bind_result< _Result, _Signature > >`, 2679
- `std::is_bind_expression< const volatile _Bind< _Signature > >`, 2680
- `std::is_bind_expression< const volatile _Bind_result< _Result, _Signature > >`, 2681
- `std::is_bind_expression< volatile _Bind< _Signature > >`, 2682
- `std::is_bind_expression< volatile _Bind_result< _Result, _Signature > >`, 2683
- `std::is_class< _Tp >`, 2684
- `std::is_compound< _Tp >`, 2685
- `std::is_const< typename >`, 2686
- `std::is_empty< _Tp >`, 2687
- `std::is_enum< _Tp >`, 2688
- `std::is_error_code_enum< _Tp >`, 2689
- `std::is_error_code_enum< future_errc >`, 2690
- `std::is_error_condition_enum< _Tp >`, 2691
- `std::is_final< _Tp >`, 2692
- `std::is_floating_point< _Tp >`, 2693
- `std::is_function< typename >`, 2693
- `std::is_fundamental< _Tp >`, 2694
- `std::is_integral< _Tp >`, 2694
- `std::is_literal_type< _Tp >`, 2695
- `std::is_lvalue_reference< typename >`, 2696
- `std::is_member_function_pointer< _Tp >`, 2697
- `std::is_member_object_pointer< _Tp >`, 2698
- `std::is_member_pointer< _Tp >`, 2699
- `std::is_null_pointer< _Tp >`, 2700
- `std::is_object< _Tp >`, 2700
- `std::is_placeholder< _Placeholder< _Num > >`, 2702
- `std::is_placeholder< _Tp >`, 2701
- `std::is_pod< _Tp >`, 2703
- `std::is_pointer< _Tp >`, 2704
- `std::is_polymorphic< _Tp >`, 2705
- `std::is_reference< _Tp >`, 2706
- `std::is_rvalue_reference< typename >`, 2706
- `std::is_scalar< _Tp >`, 2707
- `std::is_standard_layout< _Tp >`, 2708
- `std::is_trivial< _Tp >`, 2709
- `std::is_union< _Tp >`, 2710
- `std::is_void< _Tp >`, 2711
- `std::is_volatile< typename >`, 2712
- `std::istream_iterator`
 - `difference_type`, 2714
 - `istream_iterator`, 2714
 - `iterator_category`, 2714
 - `pointer`, 2714
 - `reference`, 2714
 - `value_type`, 2714
- `std::istream_iterator< _Tp, _CharT, _Traits, _Dist >`, 2713
- `std::istreambuf_iterator`
 - `char_type`, 2716
 - `difference_type`, 2716
 - `equal`, 2718
 - `int_type`, 2716
 - `istream_type`, 2717
 - `istreambuf_iterator`, 2718
 - `iterator_category`, 2717
 - `operator*`, 2718
 - `operator++`, 2718, 2719
 - `pointer`, 2717
 - `reference`, 2717
 - `streambuf_type`, 2717
 - `traits_type`, 2717
 - `value_type`, 2717
- `std::istreambuf_iterator< _CharT, _Traits >`, 2715
- `std::iterator`
 - `difference_type`, 2720
 - `iterator_category`, 2720
 - `pointer`, 2720
 - `reference`, 2720
 - `value_type`, 2720
- `std::iterator< _Category, _Tp, _Distance, _Pointer, _Reference >`, 2719
- `std::iterator_traits< _Tp * >`, 2720
- `std::iterator_traits< const _Tp * >`, 2721
- `std::length_error`, 2722
 - `what`, 2722
- `std::less`
 - `first_argument_type`, 2723
 - `result_type`, 2723
 - `second_argument_type`, 2724
- `std::less< _Tp >`, 2723
- `std::less< void >`, 2724
- `std::less_equal`
 - `first_argument_type`, 2725
 - `result_type`, 2725
 - `second_argument_type`, 2726
- `std::less_equal< _Tp >`, 2725
- `std::less_equal< void >`, 2726
- `std::linear_congruential_engine`
 - `discard`, 2728
 - `increment`, 2731
 - `linear_congruential_engine`, 2728
 - `max`, 2728
 - `min`, 2729
 - `modulus`, 2731
 - `multiplier`, 2731
 - `operator<<`, 2730
 - `operator>>`, 2730
 - `operator()`, 2729
 - `operator==`, 2730
 - `result_type`, 2728
 - `seed`, 2729

- `std::linear_congruential_engine< _UIntType, __a, __c, __m >`, 2726
- `std::list`
 - `_M_create_node`, 2738
 - `assign`, 2738, 2739
 - `back`, 2739, 2740
 - `begin`, 2740
 - `cbegin`, 2740
 - `cend`, 2740
 - `clear`, 2740
 - `crbegin`, 2741
 - `crend`, 2741
 - `emplace`, 2741
 - `empty`, 2741
 - `end`, 2741, 2742
 - `erase`, 2742
 - `front`, 2743
 - `get_allocator`, 2743
 - `insert`, 2743–2745
 - `list`, 2736–2738
 - `max_size`, 2745
 - `merge`, 2746
 - `operator=`, 2746, 2747
 - `pop_back`, 2748
 - `pop_front`, 2748
 - `push_back`, 2748
 - `push_front`, 2748
 - `rbegin`, 2749
 - `remove`, 2749
 - `remove_if`, 2749
 - `rend`, 2750
 - `resize`, 2750
 - `reverse`, 2751
 - `size`, 2751
 - `sort`, 2751
 - `splice`, 2751–2753
 - `swap`, 2753
 - `unique`, 2753, 2754
- `std::list< _Tp, _Alloc >`, 2732
- `std::locale`, 2754
 - `~locale`, 2760
 - `all`, 2764
 - `category`, 2756
 - `classic`, 2760
 - `collate`, 2764
 - `combine`, 2760
 - `ctype`, 2764
 - `global`, 2760
 - `has_facet`, 2763
 - `locale`, 2756, 2758, 2759
 - `messages`, 2764
 - `monetary`, 2764
 - `name`, 2761
 - `none`, 2765
 - `numeric`, 2765
 - `operator!=`, 2761
 - `operator()`, 2761
 - `operator=`, 2762
 - `operator==`, 2762
 - `time`, 2765
 - `use_facet`, 2763
- `std::locale::facet`, 2766
 - `~facet`, 2767
 - `facet`, 2767
- `std::locale::id`, 2768
 - `has_facet`, 2768
 - `id`, 2768
 - `use_facet`, 2769
- `std::lock_guard< _Mutex >`, 2770
- `std::logic_error`, 2770
 - `logic_error`, 2771
 - `what`, 2771
- `std::logical_and`
 - `first_argument_type`, 2772
 - `result_type`, 2772
 - `second_argument_type`, 2772
- `std::logical_and< _Tp >`, 2771
- `std::logical_and< void >`, 2773
- `std::logical_not`
 - `argument_type`, 2774
 - `result_type`, 2774
- `std::logical_not< _Tp >`, 2773
- `std::logical_not< void >`, 2774
- `std::logical_or`
 - `first_argument_type`, 2776
 - `result_type`, 2776
 - `second_argument_type`, 2776
- `std::logical_or< _Tp >`, 2775
- `std::logical_or< void >`, 2776
- `std::lognormal_distribution`
 - `max`, 2778
 - `min`, 2778
 - `operator<<`, 2779
 - `operator>>`, 2781
 - `operator()`, 2779
 - `operator==`, 2781
 - `param`, 2779
 - `reset`, 2779
 - `result_type`, 2778
- `std::lognormal_distribution< _RealType >`, 2777
- `std::lognormal_distribution< _RealType >::param_type`, 2781
- `std::map`
 - `at`, 2788
 - `begin`, 2789
 - `cbegin`, 2789
 - `cend`, 2789
 - `clear`, 2789

- count, [2789](#), [2790](#)
- crbegin, [2790](#)
- crend, [2790](#)
- emplace, [2790](#)
- emplace_hint, [2792](#)
- empty, [2792](#)
- end, [2793](#)
- equal_range, [2793](#)–[2795](#)
- erase, [2795](#), [2796](#)
- find, [2797](#), [2798](#)
- get_allocator, [2798](#)
- insert, [2798](#), [2800](#), [2801](#)
- key_comp, [2801](#)
- lower_bound, [2801](#)–[2803](#)
- map, [2785](#)–[2788](#)
- max_size, [2803](#)
- operator=, [2803](#), [2804](#)
- operator[], [2804](#)
- rbegin, [2805](#)
- rend, [2805](#)
- size, [2805](#)
- swap, [2805](#)
- upper_bound, [2806](#), [2807](#)
- value_comp, [2807](#)
- std::map< _Key, _Tp, _Compare, _Alloc >, [2782](#)
- std::mask_array< _Tp >, [2808](#)
- std::match_results
 - ~match_results, [2814](#)
 - begin, [2814](#)
 - cbegin, [2814](#)
 - cend, [2814](#)
 - empty, [2814](#)
 - end, [2815](#)
 - format, [2815](#)
 - get_allocator, [2816](#)
 - length, [2816](#)
 - match_results, [2814](#)
 - max_size, [2816](#)
 - operator=, [2816](#), [2817](#)
 - operator[], [2817](#)
 - position, [2817](#)
 - prefix, [2817](#)
 - ready, [2818](#)
 - size, [2818](#)
 - str, [2818](#)
 - suffix, [2819](#)
 - swap, [2819](#)
- std::match_results< _Bi_iter, _Alloc >, [2809](#)
- std::mem_fun1_ref_t
 - first_argument_type, [2820](#)
 - result_type, [2820](#)
 - second_argument_type, [2821](#)
- std::mem_fun1_ref_t< _Ret, _Tp, _Arg >, [2820](#)
- std::mem_fun1_t
 - first_argument_type, [2822](#)
 - result_type, [2822](#)
 - second_argument_type, [2822](#)
- std::mem_fun1_t< _Ret, _Tp, _Arg >, [2821](#)
- std::mem_fun_ref_t
 - argument_type, [2823](#)
 - result_type, [2823](#)
- std::mem_fun_ref_t< _Ret, _Tp >, [2822](#)
- std::mem_fun_t
 - argument_type, [2824](#)
 - result_type, [2824](#)
- std::mem_fun_t< _Ret, _Tp >, [2824](#)
- std::mersenne_twister_engine
 - discard, [2827](#)
 - max, [2828](#)
 - mersenne_twister_engine, [2827](#)
 - min, [2828](#)
 - operator<<, [2828](#)
 - operator>>, [2829](#)
 - operator==, [2829](#)
 - result_type, [2827](#)
- std::mersenne_twister_engine< _UIntType, __w, __n, __←
 _m, __r, __a, __u, __d, __s, __b, __t, __c, __l,
 __f >, [2825](#)
- std::messages
 - ~messages, [2832](#)
 - char_type, [2832](#)
 - do_get, [2833](#)
 - id, [2833](#)
 - messages, [2832](#)
 - string_type, [2832](#)
- std::messages< _CharT >, [2830](#)
- std::messages_base, [2833](#)
- std::messages_byname
 - do_get, [2836](#)
 - id, [2836](#)
- std::messages_byname< _CharT >, [2834](#)
- std::minus
 - first_argument_type, [2837](#)
 - result_type, [2837](#)
 - second_argument_type, [2837](#)
- std::minus< _Tp >, [2836](#)
- std::minus< void >, [2838](#)
- std::modulus
 - first_argument_type, [2839](#)
 - result_type, [2839](#)
 - second_argument_type, [2839](#)
- std::modulus< _Tp >, [2838](#)
- std::modulus< void >, [2840](#)
- std::money_base, [2840](#)
- std::money_get
 - ~money_get, [2844](#)
 - char_type, [2843](#)
 - do_get, [2844](#)

- get, [2844](#), [2845](#)
- id, [2846](#)
- iter_type, [2843](#)
- money_get, [2843](#)
- string_type, [2843](#)
- std::money_get< _CharT, _InIter >, [2841](#)
- std::money_put
 - ~money_put, [2848](#)
 - char_type, [2848](#)
 - do_put, [2849](#)
 - id, [2851](#)
 - iter_type, [2848](#)
 - money_put, [2848](#)
 - put, [2850](#)
 - string_type, [2848](#)
- std::money_put< _CharT, _OutIter >, [2846](#)
- std::moneypunct
 - ~moneypunct, [2855](#)
 - char_type, [2853](#)
 - curr_symbol, [2855](#)
 - decimal_point, [2855](#)
 - do_curr_symbol, [2855](#)
 - do_decimal_point, [2855](#)
 - do_frac_digits, [2856](#)
 - do_grouping, [2856](#)
 - do_neg_format, [2856](#)
 - do_negative_sign, [2857](#)
 - do_pos_format, [2857](#)
 - do_positive_sign, [2857](#)
 - do_thousands_sep, [2858](#)
 - frac_digits, [2858](#)
 - grouping, [2858](#)
 - id, [2861](#)
 - intl, [2861](#)
 - moneypunct, [2854](#)
 - neg_format, [2859](#)
 - negative_sign, [2859](#)
 - pos_format, [2860](#)
 - positive_sign, [2860](#)
 - string_type, [2853](#)
 - thousands_sep, [2860](#)
- std::moneypunct< _CharT, _Intl >, [2851](#)
- std::moneypunct_byname
 - curr_symbol, [2863](#)
 - decimal_point, [2863](#)
 - do_curr_symbol, [2864](#)
 - do_decimal_point, [2864](#)
 - do_frac_digits, [2864](#)
 - do_grouping, [2865](#)
 - do_neg_format, [2865](#)
 - do_negative_sign, [2865](#)
 - do_pos_format, [2866](#)
 - do_positive_sign, [2866](#)
 - do_thousands_sep, [2866](#)
- frac_digits, [2867](#)
- grouping, [2867](#)
- id, [2870](#)
- neg_format, [2867](#)
- negative_sign, [2868](#)
- pos_format, [2868](#)
- positive_sign, [2869](#)
- thousands_sep, [2869](#)
- std::moneypunct_byname< _CharT, _Intl >, [2861](#)
- std::move_iterator< _Iterator >, [2870](#)
- std::multimap
 - begin, [2877](#)
 - cbegin, [2877](#)
 - cend, [2878](#)
 - clear, [2878](#)
 - count, [2878](#)
 - crbegin, [2879](#)
 - crend, [2879](#)
 - emplace, [2879](#)
 - emplace_hint, [2879](#)
 - empty, [2880](#)
 - end, [2880](#)
 - equal_range, [2880–2882](#)
 - erase, [2882](#), [2883](#)
 - find, [2884](#), [2885](#)
 - get_allocator, [2885](#)
 - insert, [2885](#), [2887](#), [2888](#)
 - key_comp, [2888](#)
 - lower_bound, [2888](#), [2889](#)
 - max_size, [2890](#)
 - multimap, [2874–2877](#)
 - operator=, [2890](#), [2891](#)
 - rbegin, [2891](#)
 - rend, [2891](#)
 - size, [2892](#)
 - swap, [2892](#)
 - upper_bound, [2892](#), [2893](#)
 - value_comp, [2894](#)
- std::multimap< _Key, _Tp, _Compare, _Alloc >, [2871](#)
- std::multiplies
 - first_argument_type, [2895](#)
 - result_type, [2895](#)
 - second_argument_type, [2895](#)
- std::multiplies< _Tp >, [2894](#)
- std::multiplies< void >, [2895](#)
- std::multiset
 - begin, [2902](#)
 - cbegin, [2902](#)
 - cend, [2902](#)
 - clear, [2902](#)
 - count, [2902](#)
 - crbegin, [2903](#)
 - crend, [2903](#)
 - emplace, [2903](#)

- emplace_hint, 2904
- empty, 2904
- end, 2904
- equal_range, 2904–2906
- erase, 2906, 2907
- find, 2908, 2909
- get_allocator, 2909
- insert, 2909–2911
- key_comp, 2911
- lower_bound, 2911, 2912
- max_size, 2913
- multiset, 2899–2901
- operator=, 2913
- rbegin, 2914
- rend, 2914
- size, 2914
- swap, 2914
- upper_bound, 2915, 2916
- value_comp, 2916
- std::multiset< _Key, _Compare, _Alloc >, 2896
- std::mutex, 2916
- std::negate
 - argument_type, 2918
 - result_type, 2918
- std::negate< _Tp >, 2917
- std::negate< void >, 2918
- std::negative_binomial_distribution
 - k, 2920
 - max, 2920
 - min, 2920
 - operator<<, 2922
 - operator>>, 2922
 - operator(), 2921
 - operator==, 2922
 - p, 2921
 - param, 2921
 - reset, 2921
 - result_type, 2920
- std::negative_binomial_distribution< _IntType >, 2919
- std::negative_binomial_distribution< _IntType >::param←_type, 2923
- std::nested_exception, 2923
- std::normal_distribution
 - max, 2926
 - mean, 2926
 - min, 2926
 - normal_distribution, 2925
 - operator<<, 2927
 - operator>>, 2927
 - operator(), 2926
 - operator==, 2927
 - param, 2926
 - reset, 2927
 - result_type, 2925
 - stddev, 2927
- std::normal_distribution< _RealType >, 2924
- std::normal_distribution< _RealType >::param_type, 2928
- std::not_equal_to
 - first_argument_type, 2929
 - result_type, 2929
 - second_argument_type, 2930
- std::not_equal_to< _Tp >, 2929
- std::not_equal_to< void >, 2930
- std::num_get
 - ~num_get, 2933
 - char_type, 2933
 - do_get, 2934–2939
 - get, 2940–2946
 - id, 2947
 - iter_type, 2933
 - num_get, 2933
- std::num_get< _CharT, _InIter >, 2931
- std::num_put
 - ~num_put, 2950
 - char_type, 2950
 - do_put, 2950–2954
 - id, 2961
 - iter_type, 2950
 - num_put, 2950
 - put, 2955–2960
- std::num_put< _CharT, _OutIter >, 2948
- std::numeric_limits
 - denorm_min, 2962
 - digits, 2964
 - digits10, 2964
 - epsilon, 2962
 - has_denorm, 2964
 - has_denorm_loss, 2964
 - has_infinity, 2964
 - has_quiet_NaN, 2965
 - has_signaling_NaN, 2965
 - infinity, 2963
 - is_bounded, 2965
 - is_exact, 2965
 - is_iec559, 2965
 - is_integer, 2965
 - is_modulo, 2965
 - is_signed, 2966
 - is_specialized, 2966
 - lowest, 2963
 - max, 2963
 - max_digits10, 2966
 - max_exponent, 2966
 - max_exponent10, 2966
 - min, 2963
 - min_exponent, 2966
 - min_exponent10, 2966

- quiet_NaN, 2963
- radix, 2967
- round_error, 2964
- round_style, 2967
- signaling_NaN, 2964
- tinyness_before, 2967
- traps, 2967
- std::numeric_limits< _Tp >, 2961
- std::numeric_limits< bool >, 2967
- std::numeric_limits< char >, 2968
- std::numeric_limits< char16_t >, 2969
- std::numeric_limits< char32_t >, 2970
- std::numeric_limits< double >, 2971
- std::numeric_limits< float >, 2972
- std::numeric_limits< int >, 2973
- std::numeric_limits< long >, 2974
- std::numeric_limits< long double >, 2975
- std::numeric_limits< long long >, 2976
- std::numeric_limits< short >, 2977
- std::numeric_limits< signed char >, 2978
- std::numeric_limits< unsigned char >, 2979
- std::numeric_limits< unsigned int >, 2980
- std::numeric_limits< unsigned long >, 2981
- std::numeric_limits< unsigned long long >, 2982
- std::numeric_limits< unsigned short >, 2983
- std::numeric_limits< wchar_t >, 2984
- std::numprint
 - ~numprint, 2989
 - char_type, 2988
 - decimal_point, 2989
 - do_decimal_point, 2989
 - do_falsename, 2989
 - do_grouping, 2990
 - do_thousands_sep, 2990
 - do_truename, 2990
 - falsename, 2991
 - grouping, 2991
 - id, 2992
 - numprint, 2988
 - string_type, 2988
 - thousands_sep, 2991
 - truename, 2992
- std::numprint< _CharT >, 2986
- std::numprint_byname
 - decimal_point, 2994
 - do_decimal_point, 2994
 - do_falsename, 2995
 - do_grouping, 2995
 - do_thousands_sep, 2995
 - do_truename, 2996
 - falsename, 2996
 - grouping, 2996
 - id, 2998
 - thousands_sep, 2997
 - truename, 2997
- std::numprint_byname< _CharT >, 2993
- std::once_flag, 2998
 - call_once, 2999
 - once_flag, 2998
 - operator=, 2999
- std::ostream_iterator
 - char_type, 3000
 - difference_type, 3000
 - iterator_category, 3000
 - operator=, 3002
 - ostream_iterator, 3001, 3002
 - ostream_type, 3000
 - pointer, 3001
 - reference, 3001
 - traits_type, 3001
 - value_type, 3001
- std::ostream_iterator< _Tp, _CharT, _Traits >, 2999
- std::ostreambuf_iterator
 - char_type, 3003
 - difference_type, 3003
 - failed, 3005
 - iterator_category, 3004
 - operator*, 3005
 - operator++, 3005
 - operator=, 3005
 - ostream_type, 3004
 - ostreambuf_iterator, 3005
 - pointer, 3004
 - reference, 3004
 - streambuf_type, 3004
 - traits_type, 3004
 - value_type, 3004
- std::ostreambuf_iterator< _CharT, _Traits >, 3002
- std::out_of_range, 3006
 - what, 3007
- std::output_iterator_tag, 3007
- std::overflow_error, 3007
 - what, 3008
- std::owner_less< _Tp >, 3008
- std::owner_less< shared_ptr< _Tp > >, 3008
 - first_argument_type, 3009
 - result_type, 3009
 - second_argument_type, 3009
- std::owner_less< weak_ptr< _Tp > >, 3010
 - first_argument_type, 3010
 - result_type, 3010
 - second_argument_type, 3010
- std::packaged_task< _Res(_ArgTypes...) >, 3011
- std::pair
 - _PCCP, 3013
 - _PCCFP, 3013
 - first, 3014
 - pair, 3014

- second, 3014
- second_type, 3013
- std::pair<_T1, _T2>, 3011
- std::piecewise_constant_distribution
 - densities, 3016
 - intervals, 3016
 - max, 3016
 - min, 3016
 - operator<<, 3017
 - operator>>, 3018
 - operator(), 3016
 - operator==, 3017
 - param, 3016, 3017
 - reset, 3017
 - result_type, 3016
- std::piecewise_constant_distribution<_RealType>, 3014
- std::piecewise_constant_distribution<_RealType>::param_type, 3018
- std::piecewise_construct_t, 3019
- std::piecewise_linear_distribution
 - densities, 3021
 - intervals, 3021
 - max, 3021
 - min, 3021
 - operator<<, 3022
 - operator>>, 3023
 - operator(), 3021
 - operator==, 3023
 - param, 3022
 - reset, 3022
 - result_type, 3021
- std::piecewise_linear_distribution<_RealType>, 3019
- std::piecewise_linear_distribution<_RealType>::param_type, 3023
- std::placeholders, 692
- std::plus
 - first_argument_type, 3025
 - result_type, 3025
 - second_argument_type, 3025
- std::plus<_Tp>, 3024
- std::pointer_to_binary_function
 - first_argument_type, 3027
 - result_type, 3027
 - second_argument_type, 3027
- std::pointer_to_binary_function<_Arg1, _Arg2, _Result>, 3026
- std::pointer_to_unary_function
 - argument_type, 3028
 - result_type, 3028
- std::pointer_to_unary_function<_Arg, _Result>, 3027
- std::pointer_traits
 - difference_type, 3029
 - element_type, 3029
 - pointer, 3029
 - rebind, 3029
- std::pointer_traits<_Ptr>, 3029
- std::pointer_traits<_Tp*>, 3030
 - difference_type, 3030
 - element_type, 3030
 - pointer, 3031
 - pointer_to, 3031
- std::poisson_distribution
 - max, 3033
 - mean, 3033
 - min, 3033
 - operator<<, 3034
 - operator>>, 3035
 - operator(), 3033
 - operator==, 3035
 - param, 3034
 - reset, 3034
 - result_type, 3033
- std::poisson_distribution<_IntType>, 3031
- std::poisson_distribution<_IntType>::param_type, 3035
- std::priority_queue
 - empty, 3039
 - pop, 3039
 - priority_queue, 3038
 - push, 3039
 - size, 3040
 - top, 3040
- std::priority_queue<_Tp, _Sequence, _Compare>, 3036
- std::promise<_Res>, 3040
- std::promise<_Res&>, 3041
- std::promise<void>, 3042
- std::queue
 - back, 3044
 - c, 3046
 - empty, 3044
 - front, 3045
 - pop, 3045
 - push, 3045
 - queue, 3044
 - size, 3046
- std::queue<_Tp, _Sequence>, 3043
- std::random_access_iterator_tag, 3046
- std::random_device
 - result_type, 3047
- std::range_error, 3048
- what, 3048
- std::ratio<_Num, _Den>, 3049
- std::ratio_equal<_R1, _R2>, 3049
- std::ratio_not_equal<_R1, _R2>, 3050
- std::raw_storage_iterator
 - difference_type, 3052
 - iterator_category, 3052
 - pointer, 3052
 - reference, 3053

- value_type, 3053
- std::raw_storage_iterator< _OutputIterator, _Tp >, 3051
- std::recursive_mutex, 3053
- std::recursive_timed_mutex, 3054
- std::reference_wrapper< _Tp >, 3054
- std::regex_constants, 693
 - __match_flag, 694
 - __polynomial, 700
 - __syntax_option, 694
- awk, 700
- basic, 700
- collate, 700
- ECMAScript, 701
- egrep, 701
- error_backref, 695
- error_badbrace, 695
- error_badrepeat, 695
- error_brace, 696
- error_brack, 696
- error_collate, 696
- error_complexity, 696
- error_ctype, 696
- error_escape, 696
- error_paren, 696
- error_range, 696
- error_space, 696
- error_stack, 696
- error_type, 695
- extended, 701
- format_default, 701
- format_first_only, 702
- format_no_copy, 702
- format_sed, 702
- grep, 703
- icase, 703
- match_any, 703
- match_continuous, 703
- match_default, 703
- match_flag_type, 695
- match_not_bol, 703
- match_not_bow, 704
- match_not_eol, 704
- match_not_eow, 704
- match_not_null, 704
- match_prev_avail, 704
- nosubs, 704
- operator[^], 697, 698
- operator[^]==, 698
- operator&, 696, 697
- operator&=, 697
- operator|, 698, 699
- operator|=, 699
- operator~, 699, 700
- optimize, 704
- syntax_option_type, 695
- std::regex_error, 3055
 - code, 3056
 - regex_error, 3056
 - what, 3056
- std::regex_iterator
 - operator!=, 3058
 - operator*, 3058
 - operator++, 3058
 - operator->, 3059
 - operator=, 3059
 - operator==, 3059
 - regex_iterator, 3057, 3058
- std::regex_iterator< _Bi_iter, _Ch_type, _Rx_traits >, 3057
- std::regex_token_iterator
 - operator!=, 3062
 - operator*, 3063
 - operator++, 3063
 - operator->, 3063
 - operator=, 3063
 - operator==, 3064
 - regex_token_iterator, 3060–3062
- std::regex_token_iterator< _Bi_iter, _Ch_type, _Rx_traits >, 3059
- std::regex_traits
 - getloc, 3065
 - imbue, 3065
 - isctype, 3066
 - length, 3066
 - lookup_classname, 3067
 - lookup_collatename, 3068
 - regex_traits, 3065
 - transform, 3068
 - transform_primary, 3069
 - translate, 3069
 - translate_nocase, 3070
 - value, 3070
- std::regex_traits< _Ch_type >, 3064
- std::rel_ops, 705
 - operator!=, 705
 - operator<=, 705
 - operator>, 706
 - operator>=, 706
- std::reverse_iterator
 - base, 3073
 - iterator_category, 3072
 - operator*, 3073
 - operator+, 3074
 - operator++, 3074
 - operator+=, 3074
 - operator-, 3075
 - operator->, 3076
 - operator--, 3075

- operator==, 3075
- operator[], 3076
- reverse_iterator, 3073
- value_type, 3072
- std::reverse_iterator<_Iterator>, 3071
- std::runtime_error, 3077
 - runtime_error, 3077
 - what, 3078
- std::scoped_allocator_adaptor<_OuterAlloc, _InnerAllocs>, 3078
- std::seed_seq, 3080
 - result_type, 3080
 - seed_seq, 3081
- std::set
 - allocator_type, 3084
 - begin, 3089
 - cbegin, 3089
 - cend, 3089
 - clear, 3089
 - const_iterator, 3084
 - const_pointer, 3084
 - const_reference, 3084
 - const_reverse_iterator, 3084
 - count, 3090, 3091
 - crbegin, 3091
 - crend, 3091
 - difference_type, 3085
 - emplace, 3091
 - emplace_hint, 3092
 - empty, 3092
 - end, 3092
 - equal_range, 3093, 3094
 - erase, 3095, 3096
 - find, 3096, 3097
 - get_allocator, 3098
 - insert, 3098, 3099
 - iterator, 3085
 - key_comp, 3100
 - key_compare, 3085
 - key_type, 3085
 - lower_bound, 3100, 3101
 - max_size, 3102
 - operator=, 3102
 - pointer, 3085
 - rbegin, 3103
 - reference, 3085
 - rend, 3103
 - reverse_iterator, 3085
 - set, 3086–3089
 - size, 3103
 - size_type, 3086
 - swap, 3103
 - upper_bound, 3103, 3104
 - value_comp, 3106
 - value_compare, 3086
 - value_type, 3086
- std::set<_Key, _Compare, _Alloc>, 3081
- std::shared_future
 - _M_get_result, 3108
 - _Ptr, 3108
 - get, 3108
 - shared_future, 3108
- std::shared_future<_Res>, 3106
- std::shared_future<_Res &>, 3109
 - _M_get_result, 3111
 - _Ptr, 3110
 - get, 3111
 - shared_future, 3110, 3111
- std::shared_future<void>, 3111
 - _M_get_result, 3113
 - _Ptr, 3113
 - shared_future, 3113
- std::shared_lock<_Mutex>, 3114
- std::shared_ptr
 - allocate_shared, 3122
 - shared_ptr, 3116–3121
- std::shared_ptr<_Tp>, 3115
- std::shared_timed_mutex, 3123
- std::shuffle_order_engine
 - base, 3126
 - discard, 3126
 - max, 3127
 - min, 3127
 - operator<<, 3128
 - operator>>, 3128
 - operator(), 3127
 - operator==, 3128
 - result_type, 3125
 - seed, 3127
 - shuffle_order_engine, 3125, 3126
- std::shuffle_order_engine<_RandomNumberEngine, _k>, 3124
- std::slice, 3129
- std::slice_array<_Tp>, 3130
- std::stack
 - empty, 3133
 - pop, 3133
 - push, 3133
 - size, 3133
 - stack, 3133
 - top, 3134
- std::stack<_Tp, _Sequence>, 3131
- std::student_t_distribution
 - max, 3136
 - min, 3136
 - operator<<, 3137
 - operator>>, 3137
 - operator(), 3136

- operator==, 3137
- param, 3136
- reset, 3137
- result_type, 3136
- std::student_t_distribution< _RealType >, 3134
- std::student_t_distribution< _RealType >::param_type, 3138
- std::sub_match
 - _PCCP, 3140
 - _PCCFP, 3140
 - compare, 3140, 3141
 - first, 3142
 - length, 3142
 - operator string_type, 3142
 - second, 3142
 - second_type, 3140
 - str, 3142
- std::sub_match< _Biter >, 3139
- std::subtract_with_carry_engine
 - discard, 3145
 - max, 3145
 - min, 3145
 - operator<<, 3146
 - operator>>, 3147
 - operator(), 3145
 - operator==, 3146
 - result_type, 3144
 - seed, 3145, 3146
 - subtract_with_carry_engine, 3144
- std::subtract_with_carry_engine< _UIntType, __w, __s, __r >, 3143
- std::system_error, 3148
 - what, 3148
- std::this_thread, 707
 - get_id, 707
 - sleep_for, 707
 - sleep_until, 707
 - yield, 707
- std::thread, 3149
 - native_handle, 3149
- std::thread::id, 3150
- std::time_base, 3150
- std::time_get
 - ~time_get, 3153
 - char_type, 3153
 - date_order, 3154
 - do_date_order, 3154
 - do_get, 3154
 - do_get_date, 3155
 - do_get_monthname, 3156
 - do_get_time, 3156
 - do_get_weekday, 3157
 - do_get_year, 3157
 - get, 3158, 3159
 - get_date, 3159
 - get_monthname, 3160
 - get_time, 3160
 - get_weekday, 3161
 - get_year, 3161
 - id, 3162
 - iter_type, 3153
 - time_get, 3153
- std::time_get< _CharT, _InIter >, 3151
- std::time_get_byname
 - date_order, 3165
 - do_date_order, 3165
 - do_get, 3165
 - do_get_date, 3166
 - do_get_monthname, 3167
 - do_get_time, 3167
 - do_get_weekday, 3168
 - do_get_year, 3168
 - get, 3169, 3170
 - get_date, 3170
 - get_monthname, 3171
 - get_time, 3171
 - get_weekday, 3172
 - get_year, 3172
 - id, 3173
- std::time_get_byname< _CharT, _InIter >, 3163
- std::time_put
 - ~time_put, 3176
 - char_type, 3175
 - do_put, 3176
 - id, 3178
 - iter_type, 3175
 - put, 3176, 3177
 - time_put, 3175
- std::time_put< _CharT, _OutIter >, 3174
- std::time_put_byname
 - do_put, 3179
 - id, 3181
 - put, 3180
- std::time_put_byname< _CharT, _OutIter >, 3178
- std::timed_mutex, 3181
- std::tr1, 708
- std::tr1::__detail, 710
- std::tr2, 711
- std::tr2::__detail, 712
- std::tr2::__dynamic_bitset_base
 - _M_w, 3184
- std::tr2::__dynamic_bitset_base< _WordT, _Alloc >, 3182
- std::tr2::__reflection_typelist< _Elements >, 3184
- std::tr2::__reflection_typelist< _First, _Rest... >, 3184
- std::tr2::__reflection_typelist<>, 3185
- std::tr2::bases< _Tp >, 3185
- std::tr2::bool_set, 3186
 - bool_set, 3187

- equals, [3187](#)
- is_emptyset, [3187](#)
- is_indeterminate, [3187](#)
- is_singleton, [3187](#)
- operator bool, [3188](#)
- std::tr2::direct_bases< _Tp >, [3188](#)
- std::tr2::dynamic_bitset
 - all, [3195](#)
 - any, [3195](#)
 - append, [3195](#)
 - clear, [3195](#)
 - count, [3195](#)
 - dynamic_bitset, [3193](#), [3194](#)
 - empty, [3196](#)
 - find_first, [3196](#)
 - find_next, [3196](#)
 - flip, [3196](#), [3197](#)
 - get_allocator, [3197](#)
 - max_size, [3197](#)
 - none, [3197](#)
 - num_blocks, [3197](#)
 - operator<<, [3198](#)
 - operator<=, [3199](#)
 - operator>>, [3199](#)
 - operator>=, [3199](#)
 - operator^=, [3200](#)
 - operator-=, [3198](#)
 - operator=, [3199](#)
 - operator&=, [3198](#)
 - operator[], [3200](#)
 - operator|=, [3201](#)
 - operator~, [3201](#)
 - push_back, [3201](#)
 - reset, [3201](#)
 - resize, [3203](#)
 - set, [3203](#)
 - size, [3203](#)
 - swap, [3204](#)
 - test, [3204](#)
 - to_string, [3204](#)
 - to_ullong, [3204](#)
 - to_ulong, [3205](#)
- std::tr2::dynamic_bitset< _WordT, _Alloc >, [3189](#)
- std::tr2::dynamic_bitset< _WordT, _Alloc >::reference, [3205](#)
- std::try_to_lock_t, [3206](#)
- std::tuple< _Elements >, [3206](#)
- std::tuple< _T1, _T2 >, [3209](#)
- std::tuple_element< 0, std::pair< _Tp1, _Tp2 > >, [3212](#)
- std::tuple_element< 0, tuple< _Head, _Tail... > >, [3212](#)
- std::tuple_element< 1, std::pair< _Tp1, _Tp2 > >, [3212](#)
- std::tuple_element< __i, tuple< _Head, _Tail... > >, [3213](#)
- std::tuple_element< _Int, _Tp >, [3211](#)
- std::tuple_element< _Int, std::__debug::array< _Tp, _Nm > >, [3214](#)
- std::tuple_element< _Int, array< _Tp, _Nm > >, [3214](#)
- std::tuple_size< _Tp >, [3215](#)
- std::tuple_size< std::__debug::array< _Tp, _Nm > >, [3215](#)
- std::tuple_size< std::pair< _Tp1, _Tp2 > >, [3216](#)
- std::tuple_size< tuple< _Elements... > >, [3217](#)
- std::tuple_size<::array< _Tp, _Nm > >, [3218](#)
- std::type_index, [3219](#)
- std::type_info, [3220](#)
 - ~type_info, [3220](#)
 - name, [3221](#)
- std::unary_function
 - argument_type, [3222](#)
 - result_type, [3222](#)
- std::unary_function< _Arg, _Result >, [3221](#)
- std::unary_negate
 - argument_type, [3223](#)
 - result_type, [3223](#)
- std::unary_negate< _Predicate >, [3222](#)
- std::underflow_error, [3224](#)
 - what, [3224](#)
- std::uniform_int_distribution
 - max, [3226](#)
 - min, [3226](#)
 - operator(), [3226](#)
 - operator==, [3227](#)
 - param, [3226](#)
 - reset, [3227](#)
 - result_type, [3226](#)
 - uniform_int_distribution, [3226](#)
- std::uniform_int_distribution< _IntType >, [3225](#)
- std::uniform_int_distribution< _IntType >::param_type, [3227](#)
- std::uniform_real_distribution
 - max, [3229](#)
 - min, [3230](#)
 - operator(), [3230](#)
 - operator==, [3231](#)
 - param, [3230](#)
 - reset, [3230](#)
 - result_type, [3229](#)
 - uniform_real_distribution, [3229](#)
- std::uniform_real_distribution< _RealType >, [3228](#)
- std::uniform_real_distribution< _RealType >::param_type, [3231](#)
- std::unique_lock< _Mutex >, [3232](#)
- std::unique_ptr
 - ~unique_ptr, [3235](#)
 - get, [3236](#)
 - get_deleter, [3236](#)
 - operator bool, [3236](#)
 - operator*, [3236](#)

- operator->, 3236
- operator=, 3236, 3237
- release, 3237
- reset, 3237
- swap, 3238
- unique_ptr, 3234, 3235
- std::unique_ptr< _Tp, _Dp >, 3233
- std::unique_ptr< _Tp[], _Dp >, 3238
 - ~unique_ptr, 3241
 - get, 3241
 - get_deleter, 3241
 - operator bool, 3241
 - operator=, 3241, 3242
 - operator[], 3242
 - release, 3242
 - reset, 3242
 - swap, 3243
 - unique_ptr, 3239, 3240
- std::unordered_map
 - allocator_type, 3246
 - at, 3251
 - begin, 3252
 - bucket_count, 3253
 - cbegin, 3253
 - cend, 3253, 3254
 - clear, 3254
 - const_iterator, 3246
 - const_local_iterator, 3246
 - const_pointer, 3247
 - const_reference, 3247
 - count, 3254
 - difference_type, 3247
 - emplace, 3254
 - emplace_hint, 3255
 - empty, 3256
 - end, 3256
 - equal_range, 3257
 - erase, 3258, 3259
 - find, 3259, 3260
 - get_allocator, 3260
 - hash_function, 3260
 - hasher, 3247
 - insert, 3260–3263
 - iterator, 3247
 - key_eq, 3263
 - key_equal, 3247
 - key_type, 3248
 - load_factor, 3263
 - local_iterator, 3248
 - mapped_type, 3248
 - max_bucket_count, 3263
 - max_load_factor, 3263, 3264
 - max_size, 3264
 - operator=, 3264
 - operator[], 3265
 - pointer, 3248
 - reference, 3248
 - rehash, 3266
 - reserve, 3266
 - size, 3266
 - size_type, 3248
 - swap, 3266
 - unordered_map, 3249, 3250
 - value_type, 3249
- std::unordered_map< _Key, _Tp, _Hash, _Pred, _Alloc >, 3243
- std::unordered_multimap
 - allocator_type, 3270
 - begin, 3275
 - bucket_count, 3276
 - cbegin, 3276
 - cend, 3276
 - clear, 3277
 - const_iterator, 3270
 - const_local_iterator, 3270
 - const_pointer, 3270
 - const_reference, 3270
 - count, 3277
 - difference_type, 3271
 - emplace, 3277
 - emplace_hint, 3278
 - empty, 3278
 - end, 3278, 3279
 - equal_range, 3279, 3280
 - erase, 3280, 3281
 - find, 3282
 - get_allocator, 3283
 - hash_function, 3283
 - hasher, 3271
 - insert, 3283–3285
 - iterator, 3271
 - key_eq, 3286
 - key_equal, 3271
 - key_type, 3271
 - load_factor, 3286
 - local_iterator, 3271
 - mapped_type, 3272
 - max_bucket_count, 3286
 - max_load_factor, 3286
 - max_size, 3286
 - operator=, 3287
 - pointer, 3272
 - reference, 3272
 - rehash, 3287
 - reserve, 3288
 - size, 3288
 - size_type, 3272
 - swap, 3288

- unordered_multimap, [3273](#), [3274](#)
- value_type, [3272](#)
- std::unordered_multimap< _Key, _Tp, _Hash, _Pred, _Alloc >, [3267](#)
- std::unordered_multiset
 - allocator_type, [3292](#)
 - begin, [3296](#), [3297](#)
 - bucket_count, [3297](#)
 - cbegin, [3297](#)
 - cend, [3298](#)
 - clear, [3298](#)
 - const_iterator, [3292](#)
 - const_local_iterator, [3292](#)
 - const_pointer, [3292](#)
 - const_reference, [3292](#)
 - count, [3298](#)
 - difference_type, [3292](#)
 - emplace, [3299](#)
 - emplace_hint, [3299](#)
 - empty, [3300](#)
 - end, [3300](#)
 - equal_range, [3301](#)
 - erase, [3302](#), [3303](#)
 - find, [3303](#), [3304](#)
 - get_allocator, [3304](#)
 - hash_function, [3304](#)
 - hasher, [3292](#)
 - insert, [3304](#), [3306](#), [3307](#)
 - iterator, [3293](#)
 - key_eq, [3308](#)
 - key_equal, [3293](#)
 - key_type, [3293](#)
 - load_factor, [3308](#)
 - local_iterator, [3293](#)
 - max_bucket_count, [3308](#)
 - max_load_factor, [3308](#)
 - max_size, [3309](#)
 - operator=, [3309](#)
 - pointer, [3293](#)
 - reference, [3293](#)
 - rehash, [3309](#)
 - reserve, [3310](#)
 - size, [3310](#)
 - size_type, [3294](#)
 - swap, [3310](#)
 - unordered_multiset, [3294–3296](#)
 - value_type, [3294](#)
- std::unordered_multiset< _Value, _Hash, _Pred, _Alloc >, [3289](#)
- std::unordered_set
 - allocator_type, [3314](#)
 - begin, [3318](#), [3320](#)
 - bucket_count, [3320](#)
 - cbegin, [3320](#)
 - cend, [3321](#)
 - clear, [3321](#)
 - const_iterator, [3314](#)
 - const_local_iterator, [3314](#)
 - const_pointer, [3314](#)
 - const_reference, [3314](#)
 - count, [3321](#)
 - difference_type, [3315](#)
 - emplace, [3322](#)
 - emplace_hint, [3322](#)
 - empty, [3323](#)
 - end, [3323](#), [3324](#)
 - equal_range, [3324](#)
 - erase, [3325](#), [3326](#)
 - find, [3327](#)
 - get_allocator, [3327](#)
 - hash_function, [3328](#)
 - hasher, [3315](#)
 - insert, [3328–3330](#)
 - iterator, [3315](#)
 - key_eq, [3330](#)
 - key_equal, [3315](#)
 - key_type, [3315](#)
 - load_factor, [3331](#)
 - local_iterator, [3315](#)
 - max_bucket_count, [3331](#)
 - max_load_factor, [3331](#)
 - max_size, [3331](#)
 - operator=, [3332](#)
 - pointer, [3316](#)
 - reference, [3316](#)
 - rehash, [3332](#)
 - reserve, [3333](#)
 - size, [3333](#)
 - size_type, [3316](#)
 - swap, [3333](#)
 - unordered_set, [3316–3318](#)
 - value_type, [3316](#)
- std::unordered_set< _Value, _Hash, _Pred, _Alloc >, [3311](#)
- std::uses_allocator< _Tp, _Alloc >, [3334](#)
- std::uses_allocator< tuple< _Types... >, _Alloc >, [3334](#)
- std::valarray
 - valarray, [3337](#)
- std::valarray< _Tp >, [3335](#)
- std::vector
 - _M_allocate_and_copy, [3344](#)
 - _M_range_check, [3344](#)
 - ~vector, [3344](#)
 - assign, [3345](#)
 - at, [3346](#)
 - back, [3347](#)
 - begin, [3347](#)
 - capacity, [3347](#)

- cbegin, 3348
- cend, 3348
- clear, 3348
- crbegin, 3348
- crend, 3348
- data, 3348
- emplace, 3348
- empty, 3349
- end, 3349
- erase, 3349, 3350
- front, 3350
- insert, 3351, 3352
- max_size, 3353
- operator=, 3353, 3354
- operator[], 3354
- pop_back, 3355
- push_back, 3355
- rbegin, 3355
- rend, 3356
- reserve, 3356
- resize, 3356, 3357
- shrink_to_fit, 3357
- size, 3357
- swap, 3357
- vector, 3341–3344
- std::vector< _Tp, _Alloc >, 3338
- std::vector< bool, _Alloc >, 3358
- std::wbuffer_convert
 - _M_buf_locale, 3377
 - _M_in_beg, 3377
 - _M_in_cur, 3378
 - _M_in_end, 3378
 - _M_out_beg, 3378
 - _M_out_cur, 3378
 - _M_out_end, 3378
 - __streambuf_type, 3364
- char_type, 3364
- eback, 3365
- egptr, 3365
- eptr, 3365
- gbump, 3365
- getloc, 3366
- gptr, 3366
- imbue, 3366
- in_avail, 3367
- int_type, 3364
- off_type, 3364
- overflow, 3367
- pbackfail, 3367
- pbase, 3368
- pbump, 3368
- pos_type, 3364
- pptr, 3369
- pubimbue, 3369
- pubseekoff, 3369
- pubseekpos, 3370
- pubsetbuf, 3370
- pubsync, 3370
- sbumpc, 3370
- seekoff, 3370
- seekpos, 3371
- setbuf, 3371
- setg, 3371
- setp, 3372
- sgetc, 3372
- sgetn, 3372
- showmanyc, 3373
- snextc, 3373
- sputbackc, 3373
- sputc, 3374
- sputn, 3374
- state, 3375
- sungetc, 3375
- sync, 3375
- traits_type, 3364
- uflow, 3375
- underflow, 3376
- wbuffer_convert, 3364
- xsgetn, 3376
- xspn, 3377
- std::wbuffer_convert< _Codecvt, _Elem, _Tr >, 3362
- std::weak_ptr< _Tp >, 3378
- std::weibull_distribution
 - a, 3381
 - b, 3381
 - max, 3381
 - min, 3381
 - operator(), 3381
 - operator==, 3383
 - param, 3381, 3382
 - reset, 3383
 - result_type, 3381
- std::weibull_distribution< _RealType >, 3379
- std::weibull_distribution< _RealType >::param_type, 3383
- std::wstring_convert
 - converted, 3386
 - from_bytes, 3386, 3387
 - state, 3387
 - to_bytes, 3387, 3388
 - wstring_convert, 3385
- std::wstring_convert< _Codecvt, _Elem, _Wide_alloc, _← Byte_alloc >, 3384
- std_mutex.h, 3750
- stdc++.h, 3750
- stddev
 - std::normal_distribution, 2927
- stdexcept, 3751

- stdio_filebuf
 - __gnu_cxx::stdio_filebuf, 873
- stdio_filebuf.h, 3751
- stdio_sync_filebuf.h, 3752
- stdlib.h, 3752
- stdtr1c++.h, 3752
- stl_algo.h, 3752
 - __rotate, 3762
- stl_algobase.h, 3763
- stl_bvector.h, 3765
- stl_construct.h, 3766
- stl_deque.h, 3767
 - _GLIBCXX_DEQUE_BUF_SIZE, 3769
- stl_function.h, 3769
- stl_heap.h, 3772
- stl_iterator.h, 3773, 3776
- stl_iterator_base_funcs.h, 3777
- stl_iterator_base_types.h, 3778
- stl_list.h, 3779
- stl_map.h, 3780
- stl_multimap.h, 3781
- stl_multiset.h, 3781
- stl_numeric.h, 3782
- stl_pair.h, 3783
- stl_queue.h, 3784
- stl_raw_storage_iter.h, 3785
- stl_relops.h, 3785
- stl_set.h, 3786
- stl_stack.h, 3787
- stl_tempbuf.h, 3787
- stl_tree.h, 3788
- stl_uninitialized.h, 3789
- stl_vector.h, 3791
- str
 - std::basic_istream, 2005, 2006
 - std::basic_ostringstream, 2124
 - std::basic_stringbuf, 2235
 - std::basic_stringstream, 2287
 - std::match_results, 2818
 - std::sub_match, 3142
- stream_iterator.h, 3792
- streambuf, 3792
 - I/O, 100
- streambuf.tcc, 3793
- streambuf_iterator.h, 3793
- streambuf_type
 - std::istreambuf_iterator, 2717
 - std::ostreambuf_iterator, 3004
- streamoff
 - std, 560
- streampos
 - std, 560
- streamsize
 - std, 561
- stride
 - Numeric Arrays, 225
- string, 3794, 3795, 3797
 - Strings, 335
- string_conversions.h, 3798
- string_type
 - std::collate, 2377
 - std::collate_byname, 2383
 - std::messages, 2832
 - std::money_get, 2843
 - std::money_put, 2848
 - std::moneypunct, 2853
 - std::numpunct, 2988
- string_view, 3798
- string_view.tcc, 3800
- stringbuf
 - I/O, 101
- stringfwd.h, 3800
- Strings, 335
 - string, 335
 - u16string, 335
 - u32string, 335
 - wstring, 336
- stringstream
 - I/O, 101
- strstream, 3801
- substr
 - __gnu_cxx::__versa_string, 792
 - std::basic_string, 2218
- subtract_with_carry_engine
 - std::subtract_with_carry_engine, 3144
- subtractive_rng
 - __gnu_cxx::subtractive_rng, 915
- suffix
 - std::match_results, 2819
- sum
 - Numeric Arrays, 225
- sungetc
 - __gnu_cxx::enc_filebuf, 830
 - __gnu_cxx::stdio_filebuf, 889
 - __gnu_cxx::stdio_sync_filebuf, 910
 - std::basic_filebuf, 1718
 - std::basic_streambuf, 2161
 - std::basic_stringbuf, 2235
 - std::wbuffer_convert, 3375
- swap
 - __gnu_cxx, 388
 - __gnu_cxx::__versa_string, 793
 - __gnu_debug::basic_string, 1008
 - __gnu_pbds::sample_probe_fn, 1353
 - __gnu_pbds::sample_range_hashing, 1355
 - __gnu_pbds::sample_ranged_hash_fn, 1356
 - __gnu_pbds::sample_resize_policy, 1359
 - __gnu_pbds::sample_resize_trigger, 1362

- __gnu_pbds::sample_size_policy, [1364](#)
- __gnu_pbds::sample_update_policy, [1367](#)
- fs_path.h, [3546](#)
- Futures, [87](#)
- Mutexes, [169](#)
- Numeric Arrays, [225](#)
- Regular Expressions, [298](#), [299](#)
- std, [639–641](#)
- std::__debug, [652](#)
- std::__profile, [679](#)
- std::basic_regex, [2144](#)
- std::basic_string, [2219](#)
- std::deque, [2503](#)
- std::experimental::fundamentals_v1::any, [2530](#)
- std::forward_list, [2575](#)
- std::function< _Res(_ArgTypes...)>, [2587](#)
- std::list, [2753](#)
- std::map, [2805](#)
- std::match_results, [2819](#)
- std::multimap, [2892](#)
- std::multiset, [2914](#)
- std::set, [3103](#)
- std::tr2::dynamic_bitset, [3204](#)
- std::unique_ptr, [3238](#)
- std::unique_ptr< _Tp[], _Dp >, [3243](#)
- std::unordered_map, [3266](#)
- std::unordered_multimap, [3288](#)
- std::unordered_multiset, [3310](#)
- std::unordered_set, [3333](#)
- std::vector, [3357](#)
- Type-safe container of any type, [345](#)
- Utilities, [358](#), [359](#)
- swap_ranges
 - Mutating, [163](#)
- sync
 - __gnu_cxx::enc_filebuf, [830](#)
 - __gnu_cxx::stdio_filebuf, [889](#)
 - __gnu_cxx::stdio_sync_filebuf, [910](#)
 - std::basic_filebuf, [1718](#)
 - std::basic_fstream, [1772](#)
 - std::basic_ifstream, [1821](#)
 - std::basic_iostream, [1908](#)
 - std::basic_istream, [1955](#)
 - std::basic_istreamstream, [2006](#)
 - std::basic_streambuf, [2161](#)
 - std::basic_stringbuf, [2236](#)
 - std::basic_stringstream, [2287](#)
 - std::wbuffer_convert, [3375](#)
- sync_with_stdio
 - std::basic_fstream, [1772](#)
 - std::basic_ifstream, [1822](#)
 - std::basic_ios, [1851](#)
 - std::basic_iostream, [1908](#)
 - std::basic_istream, [1956](#)
- std::basic_istreamstream, [2006](#)
- std::basic_ofstream, [2047](#)
- std::basic_ostream, [2084](#)
- std::basic_ostreamstream, [2124](#)
- std::basic_stringstream, [2288](#)
- std::ios_base, [2663](#)
- syntax_option_type
 - std::regex_constants, [695](#)
- synth_access_traits
 - __gnu_pbds::detail::trie_traits< Key, Mapped, _↵
ATraits, Node_Update, pat_trie_tag, _Alloc >, [1311](#)
 - __gnu_pbds::detail::trie_traits< Key, null_type, _↵
ATraits, Node_Update, pat_trie_tag, _Alloc >, [1312](#)
- synth_access_traits.hpp, [3801](#)
- system_error, [3802](#), [3803](#)
- t
 - std::binomial_distribution, [2312](#)
- TLB_size
 - __gnu_parallel::_Settings, [1109](#)
- table
 - std::ctype< char >, [2422](#)
 - std::ctype_byname< char >, [2466](#)
- table_size
 - std::ctype< char >, [2425](#)
 - std::ctype_byname< char >, [2469](#)
- tag_and_trait.hpp, [3803](#)
- Tags, [337](#)
 - trivial_iterator_difference_type, [337](#)
- tags.h, [3805](#)
- tan
 - Complex Numbers, [56](#)
- tanh
 - Complex Numbers, [56](#)
- target
 - std::function< _Res(_ArgTypes...)>, [2587](#), [2588](#)
- target_type
 - std::function< _Res(_ArgTypes...)>, [2588](#)
- tellg
 - std::basic_fstream, [1773](#)
 - std::basic_ifstream, [1822](#)
 - std::basic_iostream, [1909](#)
 - std::basic_istream, [1956](#)
 - std::basic_istreamstream, [2007](#)
 - std::basic_stringstream, [2288](#)
- tellp
 - std::basic_fstream, [1773](#)
 - std::basic_iostream, [1909](#)
 - std::basic_ofstream, [2047](#)
 - std::basic_ostream, [2084](#)
 - std::basic_ostreamstream, [2125](#)
 - std::basic_stringstream, [2289](#)

- temporary_buffer
 - __gnu_cxx::temporary_buffer, 917
- terminate
 - Exceptions, 75
- terminate_handler
 - Exceptions, 74
- test
 - std::bitset, 2325
 - std::tr2::dynamic_bitset, 3204
- tgmath.h, 3805
- thin_heap_.hpp, 3805
- thousands_sep
 - std::moneypunct, 2860
 - std::moneypunct_byname, 2869
 - std::numpunct, 2991
 - std::numpunct_byname, 2997
- thread, 3806
- Threads, 338
- throw_allocator.h, 3807
- throw_with_nested
 - Exceptions, 75
- tie
 - std::basic_fstream, 1773, 1774
 - std::basic_ifstream, 1823
 - std::basic_ios, 1851
 - std::basic_iostream, 1909, 1910
 - std::basic_istream, 1957
 - std::basic_istreamstream, 2007
 - std::basic_ofstream, 2047, 2048
 - std::basic_ostream, 2085
 - std::basic_ostreamstream, 2125
 - std::basic_stringstream, 2289
 - Utilities, 359
- Time, 339
- time
 - std::locale, 2765
- time_get
 - std::time_get, 3153
- time_members.h, 3808
- time_point_cast
 - std::chrono, 682
- time_put
 - std::time_put, 3175
- tinyness_before
 - std::__numeric_limits_base, 1571
 - std::numeric_limits, 2967
- to_bytes
 - std::wstring_convert, 3387, 3388
- to_string
 - std::bitset, 2325
 - std::tr2::dynamic_bitset, 3204
- to_ullong
 - std::tr2::dynamic_bitset, 3204
- to_ulong
 - std::bitset, 2325
 - std::tr2::dynamic_bitset, 3205
- tolower
 - std, 641
 - std::__ctype_abstract_base, 1445, 1446
 - std::ctype, 2409
 - std::ctype< char >, 2422, 2423
 - std::ctype< wchar_t >, 2438, 2439
 - std::ctype_byname, 2453, 2454
 - std::ctype_byname< char >, 2466, 2467
- top
 - std::priority_queue, 3040
 - std::stack, 3134
- toupper
 - std, 642
 - std::__ctype_abstract_base, 1446, 1447
 - std::ctype, 2410
 - std::ctype< char >, 2423, 2424
 - std::ctype< wchar_t >, 2439, 2440
 - std::ctype_byname, 2454, 2455
 - std::ctype_byname< char >, 2467, 2468
- trace_fn_imps.hpp, 3809, 3810
- Traits, 340
- traits.hpp, 3810–3812
- traits_type
 - std::basic_ios, 1839
 - std::basic_istream::sentry, 1967
 - std::basic_streambuf, 2148
 - std::istreambuf_iterator, 2717
 - std::ostream_iterator, 3001
 - std::ostreambuf_iterator, 3004
 - std::wbuffer_convert, 3364
- transform
 - Mutating, 164
 - std::collate, 2380
 - std::collate_byname, 2385
 - std::regex_traits, 3068
- transform_minimal_n
 - __gnu_parallel::_Settings, 1109
- transform_primary
 - std::regex_traits, 3069
- translate
 - std::regex_traits, 3069
- translate_nocase
 - std::regex_traits, 3070
- traps
 - std::__numeric_limits_base, 1571
 - std::numeric_limits, 2967
- tree
 - __gnu_pbds::tree, 1373
- tree_policy.hpp, 3812
- tree_trace_base.hpp, 3813
- trie
 - __gnu_pbds::trie, 1378

- trie_policy.hpp, [3813](#)
- trie_policy_base.hpp, [3813](#)
- trie_string_access_traits_imp.hpp, [3814](#)
- trivial_iterator_difference_type
 - Tags, [337](#)
- true_type
 - Metaprogramming, [147](#)
- trunename
 - std::numpunct, [2992](#)
 - std::numpunct_byname, [2997](#)
- trunc
 - std::basic_fstream, [1783](#)
 - std::basic_ifstream, [1832](#)
 - std::basic_ios, [1859](#)
 - std::basic_iostream, [1919](#)
 - std::basic_istream, [1966](#)
 - std::basic_istreamstream, [2016](#)
 - std::basic_ofstream, [2056](#)
 - std::basic_ostream, [2094](#)
 - std::basic_ostreamstream, [2134](#)
 - std::basic_stringstream, [2299](#)
 - std::ios_base, [2670](#)
- try_lock
 - std, [642](#)
- try_to_lock
 - Mutexes, [169](#)
- tuple, [3814](#), [3816](#)
- tuple_cat
 - Utilities, [359](#)
- type
 - __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, cc_hash_tag, Policy_TI >, [1195](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, gp_hash_tag, Policy_TI >, [1196](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, list_update_tag, Policy_TI >, [1197](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, ov_tree_tag, Policy_TI >, [1197](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, rb_tree_tag, Policy_TI >, [1198](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, Mapped, _Alloc, splay_tree_tag, Policy_TI >, [1199](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, cc_hash_tag, Policy_TI >, [1200](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, gp_hash_tag, Policy_TI >, [1200](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, list_update_tag, Policy_TI >, [1201](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, ov_tree_tag, Policy_TI >, [1202](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, pat_trie_tag, Policy_TI >, [1202](#)
 - __gnu_pbds::detail::container_base_dispatch< Key, null_type, _Alloc, splay_tree_tag, Policy_TI >, [1203](#)
 - __gnu_pbds::detail::container_base_dispatch< _V← Tp, Cmp_Fn, _Alloc, binary_heap_tag, null_type >, [1192](#)
 - __gnu_pbds::detail::container_base_dispatch< _V← Tp, Cmp_Fn, _Alloc, binomial_heap_tag, null←_type >, [1193](#)
 - __gnu_pbds::detail::container_base_dispatch< _V← Tp, Cmp_Fn, _Alloc, pairing_heap_tag, null←_type >, [1193](#)
 - __gnu_pbds::detail::container_base_dispatch< _← VTp, Cmp_Fn, _Alloc, rc_binomial_heap_tag, null_type >, [1194](#)
 - __gnu_pbds::detail::container_base_dispatch< _V← Tp, Cmp_Fn, _Alloc, thin_heap_tag, null_type >, [1195](#)
 - __gnu_pbds::detail::default_comb_hash_fn, [1204](#)
 - __gnu_pbds::detail::default_eq_fn, [1205](#)
 - __gnu_pbds::detail::default_hash_fn, [1205](#)
 - __gnu_pbds::detail::default_probe_fn, [1206](#)
 - __gnu_pbds::detail::default_resize_policy, [1206](#)
 - __gnu_pbds::detail::default_trie_access_traits< std::basic_string< Char, Char_Traits, std←::allocator< char > > >, [1207](#)
 - __gnu_pbds::detail::default_update_policy, [1208](#)
 - __gnu_pbds::detail::entry_cmp< _VTp, Cmp_Fn, ←_Alloc, true >, [1210](#)
 - std::experimental::fundamentals_v1::any, [2530](#)
- Type-safe container of any type, [342](#)
 - any_cast, [343–345](#)
 - swap, [345](#)
- type_traits, [3817–3819](#)
- type_traits.h, [3822](#)
- type_utils.hpp, [3823](#)
- typeidindex, [3823](#)
- typeidinfo, [3824](#)
- typelist.h, [3824](#)
- types.h, [3825](#)
- types_traits.hpp, [3826](#)
- u16streampos
 - std, [561](#)
- u16string
 - Strings, [335](#)
- u32streampos

- std, 561
- u32string
 - Strings, 335
- u8path
 - fs_path.h, 3547
- uflow
 - __gnu_cxx::enc_filebuf, 830
 - __gnu_cxx::stdio_filebuf, 889
 - __gnu_cxx::stdio_sync_filebuf, 910
 - std::basic_filebuf, 1718
 - std::basic_streambuf, 2161
 - std::basic_stringbuf, 2236
 - std::wbuffer_convert, 3375
- uncaught_exception
 - Exceptions, 75
- uncaught_exceptions
 - Exceptions, 75
- underflow
 - __gnu_cxx::enc_filebuf, 831
 - __gnu_cxx::stdio_filebuf, 890
 - __gnu_cxx::stdio_sync_filebuf, 911
 - std::basic_filebuf, 1719
 - std::basic_streambuf, 2162
 - std::basic_stringbuf, 2236
 - std::wbuffer_convert, 3376
- unexpected
 - Exceptions, 75
- unexpected_handler
 - Exceptions, 74
- unget
 - std::basic_fstream, 1774
 - std::basic_ifstream, 1823
 - std::basic_iostream, 1910
 - std::basic_istream, 1957
 - std::basic_istreamstream, 2008
 - std::basic_stringstream, 2290
- Uniform Distributions, 346
 - operator!=, 346
 - operator<<, 347
 - operator>>, 347, 348
- uniform_int_dist.h, 3827
- uniform_int_distribution
 - std::uniform_int_distribution, 3226
- uniform_real_distribution
 - std::uniform_real_distribution, 3229
- uninitialized_copy
 - std, 642
- uninitialized_copy_n
 - SGL, 308
 - std, 643
- uninitialized_fill
 - std, 643
- uninitialized_fill_n
 - std, 644
- unique
 - Mutating, 165
 - std::forward_list, 2575, 2576
 - std::list, 2753, 2754
- unique_copy
 - Mutating, 166
- unique_copy.h, 3827
- unique_copy_minimal_n
 - __gnu_parallel::_Settings, 1109
- unique_ptr
 - std::unique_ptr, 3234, 3235
 - std::unique_ptr<_Tp[], _Dp>, 3239, 3240
- unique_ptr.h, 3828
- unitbuf
 - std, 644
 - std::basic_fstream, 1783
 - std::basic_ifstream, 1832
 - std::basic_ios, 1859
 - std::basic_iostream, 1919
 - std::basic_istream, 1966
 - std::basic_istreamstream, 2016
 - std::basic_ofstream, 2056
 - std::basic_ostream, 2094
 - std::basic_ostreamstream, 2134
 - std::basic_stringstream, 2299
 - std::ios_base, 2670
- Unordered Associative, 349
- unordered_base.h, 3829
- unordered_map, 3829–3832
 - std::unordered_map, 3249, 3250
- unordered_map.h, 3832
- unordered_multimap
 - std::unordered_multimap, 3273, 3274
- unordered_multiset
 - std::unordered_multiset, 3294–3296
- unordered_set, 3833–3836
 - std::unordered_set, 3316–3318
- unordered_set.h, 3836
- unsetf
 - std::basic_fstream, 1775
 - std::basic_ifstream, 1824
 - std::basic_ios, 1852
 - std::basic_iostream, 1911
 - std::basic_istream, 1958
 - std::basic_istreamstream, 2008
 - std::basic_ofstream, 2048
 - std::basic_ostream, 2086
 - std::basic_ostreamstream, 2126
 - std::basic_stringstream, 2290
 - std::ios_base, 2663
- unshift
 - std::__codecvt_abstract_base, 1433
 - std::codecvt, 2347

- std::codecvt< _InternT, _ExternT, encoding_state >, 2351
- std::codecvt< char, char, mbstate_t >, 2355
- std::codecvt< char16_t, char, mbstate_t >, 2360
- std::codecvt< char32_t, char, mbstate_t >, 2364
- std::codecvt< wchar_t, char, mbstate_t >, 2368
- std::codecvt_byname, 2374
- update_fn_imps.hpp, 3837
- upper_bound
 - Binary Search, 39, 40
 - std::map, 2806, 2807
 - std::multimap, 2892, 2893
 - std::multiset, 2915, 2916
 - std::set, 3103, 3104
- uppercase
 - std, 644
 - std::basic_fstream, 1784
 - std::basic_ifstream, 1832
 - std::basic_ios, 1860
 - std::basic_iostream, 1920
 - std::basic_istream, 1966
 - std::basic_istream, 2016
 - std::basic_ofstream, 2057
 - std::basic_ostream, 2094
 - std::basic_ostringstream, 2134
 - std::basic_stringstream, 2299
 - std::ios_base, 2671
- use_facet
 - Locales, 113
 - std::locale, 2763
 - std::locale::id, 2769
- Utilities, 350
 - __addressof, 353
 - addressof, 353
 - forward, 353, 354
 - get, 354, 355
 - make_pair, 355
 - move, 356
 - move_if_noexcept, 356
 - operator!=, 357
 - operator<, 357
 - operator<=, 357
 - operator>, 357
 - operator>=, 358
 - operator==, 357
 - piecewise_construct, 359
 - swap, 358, 359
 - tie, 359
 - tuple_cat, 359
- utility, 3838, 3839
- valarray, 3840
 - Numeric Arrays, 202, 203
 - std::valarray, 3337
- valarray_after.h, 3843
- valarray_array.h, 3853
- valarray_array.tcc, 3861
- valarray_before.h, 3862
- valid_prefix
 - __gnu_pbds::detail::pat_trie_base::_Node_citer, 1263
 - __gnu_pbds::detail::pat_trie_base::_Node_iter, 1266
- value
 - std::regex_traits, 3070
- value_comp
 - std::map, 2807
 - std::multimap, 2894
 - std::multiset, 2916
 - std::set, 3106
- value_compare
 - std::set, 3086
- value_type
 - __gnu_pbds::detail::bin_search_tree_const_node_↵ it_, 1161
 - __gnu_pbds::detail::bin_search_tree_node_it_, 1166
 - __gnu_pbds::detail::binary_heap_const_iterator_, 1174
 - __gnu_pbds::detail::binary_heap_point_const_↵ iterator_, 1178
 - __gnu_pbds::detail::left_child_next_sibling_heap_↵ const_iterator_, 1224
 - __gnu_pbds::detail::left_child_next_sibling_heap_↵ node_point_const_iterator_, 1228
 - const_iterator_, 1411
 - iterator_, 1415
 - point_const_iterator_, 1419
 - point_iterator_, 1422
 - std::allocator_traits, 1651
 - std::allocator_traits< allocator< _Tp > >, 1656
 - std::back_insert_iterator, 1691
 - std::complex, 2387
 - std::front_insert_iterator, 2580
 - std::insert_iterator, 2648
 - std::istream_iterator, 2714
 - std::istreambuf_iterator, 2717
 - std::iterator, 2720
 - std::ostream_iterator, 3001
 - std::ostreambuf_iterator, 3004
 - std::raw_storage_iterator, 3053
 - std::reverse_iterator, 3072
 - std::set, 3086
 - std::unordered_map, 3249
 - std::unordered_multimap, 3272
 - std::unordered_multiset, 3294
 - std::unordered_set, 3316
- vector, 3863, 3864
 - std::__debug::vector, 1510
 - std::vector, 3341–3344

- vector.tcc, [3865](#)
- void_pointer
 - [__gnu_cxx::__alloc_traits](#), [716](#)
 - [std::allocator_traits](#), [1651](#)
 - [std::allocator_traits< allocator< _Tp > >](#), [1656](#)
- vstring.h, [3865](#)
- vstring.tcc, [3868](#)
- vstring_fwd.h, [3869](#)
- vstring_util.h, [3869](#)
- wbuffer_convert
 - [std::wbuffer_convert](#), [3364](#)
- wcerr
 - [std](#), [647](#)
- wcin
 - [std](#), [647](#)
- wclog
 - [std](#), [647](#)
- wcout
 - [std](#), [647](#)
- wregex_token_iterator
 - Regular Expressions, [269](#)
- wcsub_match
 - Regular Expressions, [270](#)
- wfilebuf
 - I/O, [101](#)
- wfstream
 - I/O, [101](#)
- what
 - [__gnu_cxx::forced_error](#), [838](#)
 - [__gnu_cxx::recursive_init_error](#), [860](#)
 - [__gnu_pbds::container_error](#), [1149](#)
 - [__gnu_pbds::insert_error](#), [1334](#)
 - [__gnu_pbds::join_error](#), [1335](#)
 - [__gnu_pbds::resize_error](#), [1352](#)
 - [std::bad_alloc](#), [1694](#)
 - [std::bad_cast](#), [1695](#)
 - [std::bad_exception](#), [1696](#)
 - [std::bad_function_call](#), [1696](#)
 - [std::bad_typeid](#), [1697](#)
 - [std::bad_weak_ptr](#), [1698](#)
 - [std::domain_error](#), [2516](#)
 - [std::exception](#), [2522](#)
 - [std::experimental::fundamentals_v1::bad_any_cast](#), [2531](#)
 - [std::experimental::fundamentals_v1::bad_optional↔_access](#), [2532](#)
 - [std::future_error](#), [2596](#)
 - [std::invalid_argument](#), [2653](#)
 - [std::ios_base::failure](#), [2672](#)
 - [std::length_error](#), [2722](#)
 - [std::logic_error](#), [2771](#)
 - [std::out_of_range](#), [3007](#)
 - [std::overflow_error](#), [3008](#)
 - [std::range_error](#), [3048](#)
 - [std::regex_error](#), [3056](#)
 - [std::runtime_error](#), [3078](#)
 - [std::system_error](#), [3148](#)
 - [std::underflow_error](#), [3224](#)
- widen
 - [std::__ctype_abstract_base](#), [1447](#), [1448](#)
 - [std::basic_fstream](#), [1775](#)
 - [std::basic_ifstream](#), [1824](#)
 - [std::basic_ios](#), [1852](#)
 - [std::basic_iostream](#), [1911](#)
 - [std::basic_istream](#), [1958](#)
 - [std::basic_istreamstream](#), [2009](#)
 - [std::basic_ofstream](#), [2048](#)
 - [std::basic_ostream](#), [2086](#)
 - [std::basic_ostreamstream](#), [2126](#)
 - [std::basic_stringstream](#), [2291](#)
 - [std::ctype](#), [2411](#)
 - [std::ctype< char >](#), [2424](#), [2425](#)
 - [std::ctype< wchar_t >](#), [2440](#)
 - [std::ctype_byname](#), [2455](#), [2456](#)
 - [std::ctype_byname< char >](#), [2468](#), [2469](#)
- width
 - [std::basic_fstream](#), [1776](#)
 - [std::basic_ifstream](#), [1825](#)
 - [std::basic_ios](#), [1852](#), [1853](#)
 - [std::basic_iostream](#), [1912](#)
 - [std::basic_istream](#), [1959](#)
 - [std::basic_istreamstream](#), [2009](#)
 - [std::basic_ofstream](#), [2049](#)
 - [std::basic_ostream](#), [2086](#), [2087](#)
 - [std::basic_ostreamstream](#), [2127](#)
 - [std::basic_stringstream](#), [2291](#)
 - [std::ios_base](#), [2664](#)
- wfstream
 - I/O, [101](#)
- wios
 - I/O, [101](#)
- wiostream
 - I/O, [101](#)
- wistream
 - I/O, [101](#)
- wistreamstream
 - I/O, [102](#)
- wofstream
 - I/O, [102](#)
- workstealing.h, [3870](#)
- wostream
 - I/O, [102](#)
- wostringstream
 - I/O, [102](#)
- wregex
 - Regular Expressions, [270](#)
- write

std::basic_fstream, [1776](#)
std::basic_iostream, [1912](#)
std::basic_ofstream, [2050](#)
std::basic_ostream, [2087](#)
std::basic_ostringstream, [2127](#)
std::basic_stringstream, [2292](#)

ws

std, [644](#)

wsregex_token_iterator

Regular Expressions, [270](#)

wssub_match

Regular Expressions, [270](#)

wstreambuf

I/O, [102](#)

wstreampos

std, [561](#)

wstring

Strings, [336](#)

wstring_convert

std::wstring_convert, [3385](#)

wstringbuf

I/O, [102](#)

wstringstream

I/O, [102](#)

xalloc

std::basic_fstream, [1777](#)
std::basic_ifstream, [1826](#)
std::basic_ios, [1853](#)
std::basic_iostream, [1913](#)
std::basic_istream, [1960](#)
std::basic_istream, [2010](#)
std::basic_ofstream, [2050](#)
std::basic_ostream, [2088](#)
std::basic_ostringstream, [2128](#)
std::basic_stringstream, [2292](#)
std::ios_base, [2664](#)

xsgetn

__gnu_cxx::enc_filebuf, [831](#)
__gnu_cxx::stdio_filebuf, [890](#)
__gnu_cxx::stdio_sync_filebuf, [911](#)
std::basic_filebuf, [1719](#)
std::basic_streambuf, [2162](#)
std::basic_stringbuf, [2237](#)
std::wbuffer_convert, [3376](#)

xspn

__gnu_cxx::enc_filebuf, [832](#)
__gnu_cxx::stdio_filebuf, [891](#)
__gnu_cxx::stdio_sync_filebuf, [912](#)
std::basic_filebuf, [1720](#)
std::basic_streambuf, [2163](#)
std::basic_stringbuf, [2238](#)
std::wbuffer_convert, [3377](#)

yield

std::this_thread, [707](#)